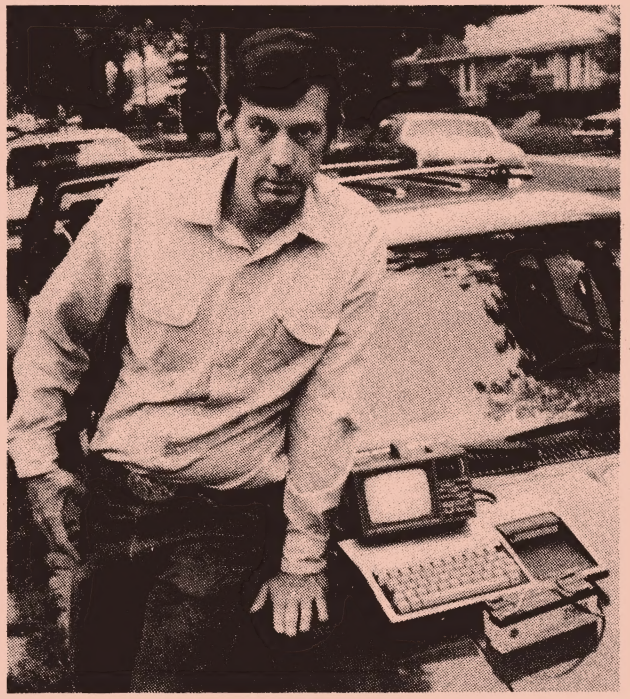
Covering the TI99/4A and the Myarc 9640

MICROpendium

Volume 6 Number 5 June 1989 $2.00

PORTABLE TI

Part I of a project to build a battery operated 99/4A with expansion memory, RAMdisk and printer port



More Calendar

1989 CALENDAR 1989

JANUARY

S	M	T	W	TH	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

FEBRUARY

S	M	T	W	TH	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

MARCH

S	M	T	W	TH	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

APRIL

S	M	T	W	TH	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

MAY

S	M	T	W	TH	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

JUNE

S	M	T	W	TH	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

JULY

S	M	T	W	TH	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

AUGUST

S	M	T	W	TH	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

SEPTEMBER

S	M	T	W	TH	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

OCTOBER

S	M	T	W	TH	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

NOVEMBER

S	M	T	W	TH	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

DECEMBER

S	M	T	W	TH	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

INSIDE

Super Savings

Tex-Comp Proudly Presents

Never Before Pricing

On Essential Software Packages for the Texas Instruments Home Computer

As part of its program of long-term support for the TI-99/4A user, Tex-Comp has purchased truck-load quantities of original TI Software that is essential to the serious & dedicated user.

In turn, Tex-Comp is passing the savings on to **YOU!**

All TI Software in this advertisement is brand new, original TI Product in factory-sealed packages and is sold with a full Texas Instruments warranty, which TI has publicly committed to.

Now is the time to buy **Key Software** at a fraction of its original cost.

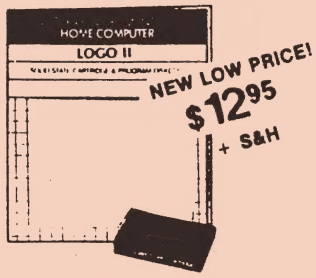
There may never be a better time than now to buy

TI Multi-Plan

Electronic Worksheet with many advanced features and built-in ease of use. Requires disk drive and controller, and 32K memory Expansion Unit. Printer and RS-232 Interface recommended. Cartridge and Disk.

TI Writer

This is a professional word processing system for the TI-99/4A. Provides the features and ease of use found in office systems. Requires disk drive. 32K Memory and Printer. Module and Disk.



Logo II

In use by educators throughout the country. Requires cassette or disk based system and 32K memory expansion.

Editor/Assembler

This is the **complete** version with manual, module, program disk and the disk version of Tombstone City as an example of assembly language programming. 32k and disk drive are required. This package will allow you to program the 99/4A in TMS 9900 Assembly Language and gives you access to all system features. Provides the fastest speed possible from the 16-bit processor!

Original TI-Extended Basic Still the BEST for Less!

A powerful, high-level programming language that expands the capability of your TI-99/4A Home Computer. Includes TI Extended BASIC module (36K bytes of preprogrammed memory) and Owner's reference manual.

Mini-Memory

This software cartridge adds memory to your system. Totals 14K of memory (6K of GROM, 4K of ROM, 4K of RAM). Mini Memory includes a built-in battery, permitting programs and data stored in RAM to be retained even if module is removed from console.

BONUS: FREE **Mini-Writer** I word processor (a $19.95 value) with Mini-Memory Purchase.

SPECIALS:

SUPER EXTENDED BASIC.ExBasic + additional commands and graphics	$49.95
EXTENDED BASIC II PLUS BY Mechatronic..ExBasic plus 44 K Bytes of preprogrammed memory plus graphics.	$44.95
TI WRITER/MULTIPLAN UPGRADE DISK (with either)...	$ 1.00
TI LOGO WORKBOOK (with LOGO).....	$ 1.00
TEACH YOURSELF EXBASIC ON DISK (with any exbasic)	$ 1.00
"INTRO TO ASSEM. LANGUAGE" (with Ed/Assem).....	$ 1.00
TERMINAL EMULATOR II (reg 9.95) with any of above	$ 4.95

TEX-COMP

America's Number One TI computer retailer

P.O. Box 33084, Granada Hills, CA 91344

VISA & MASTERCARD HOLDERS CALL DIRECT

VISA MasterCard

(818) 366-6631

24 HOURS A DAY

7 Days a Week!

TERMS: All prices FOB Los Angeles. For fastest service use cashiers check, or money order. Add 3% shipping and handling ($3.00 minimum). East of Mississippi, 4% (free shipping on all software orders over $100.00! COD to be paid by cash or certified check. All TI products are sold with the original manufacturer's guarantee only (sent on request). Prices and availability subject to change without notice. We reserve the right to limit quantities.

NOTE: Payment in full must accompany all orders. Credit Card, Company Check or Money Order for immediate shipment. Personal checks require up to 4 weeks to clear. California orders add 6½% sales tax.

Contents

MICROpendium

MICROpendium (ISSN 10432299) is published monthly for $20 per year by Burns-Koloen Communications Inc., 16606 Terrace Dr., Austin, TX 78728-1156. Second-class postage paid at Austin, Texas, and additional mailing offices. POSTMASTER: Send address changes to MICROpendium, P.O. Box 1343, Round Rock, TX 78680-1343.

Display advertising deadlines and rates are available upon request.

All correspondence should be mailed to MICROpendium at P.O. Box 1343, Round Rock, TX 78680. We cannot take responsibility for unsolicited manuscripts but will give consideration to anything sent to the above address. Manuscripts will be returned only if a self-addressed stamped envelope is included.

Foreign subscriptions are $25.25 (Mexico); $27.50 (Canada) $25.00, surface mail to other countries; $37 airmail to other countries.

All editions of MICROpendium are mailed from the Round Rock (Texas) Post Office. Mailing address: P.O. Box 1343, Round Rock TX 78680
Telephone: (512) 255-1512
Source: TI4596
CompuServe: 75156,3270
Delphi TI NET: MICROPENDIUM
GEnie: J.Koloen

John Koloen.....**Publisher**
Laura Burns.....**Editor**

Programming conventions

Here are some tips to help you when entering programs from MICROpendium:
1. All BASIC and Extended BASIC programs are run through Checksum, the numbers that follow exclamation at the end of each program line. Do not enter these numbers or exclamation points. Checksum was published in the October 1987 edition.
2. Long XBASIC lines are entered by inputting until the screen stops accepting characters, pressing Enter, pressing FCTN REDO, cursoring to the end of the line and continuing input.

Page Pro 99

Page Pro 99 is the remarkable new page-making program for your TI-99/4A or Myarc Geneve that does something no other publishing program can claim - let you see the page on the screen exactly as it will appear on paper, before it is printed.

Page Pro 99 is fast (written in assembly), easy to use (with on-screen prompts, a TI-Writer like editor, etc.), and powerful (with dozens of features, including many unique to this program). It will let you use standard TI-Artist pictures and fonts on your page. It will work on any system from a singe drive 99/4A to a Geneve with hard disk drives. It includes thorough documentation. It is very dependable and well-tested (as it should be after 2 years of development). Finally, it produces beautiful pages.

Because it is "what-you-see-is-what-you-get" it is easy-to-use. Because it includes numerous examples, a good collection of starter fonts and pictures, and an in-depth tutorial you can use it straight out of the box. Because it is by Asgard Software, it has the support that it, and you deserve. Who can ask for more?

PAGE PRO TIMES

VOL.1 NO.1 SUMMER 1989

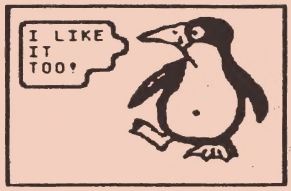
PAGE PRO 99 RELEASED!

Rockville, MD- Asgard Software demonstrated its long awaited Page Pro 99 at a press conference today to thunderous applause. After the preliminaries, the President (and chief bottle-washer) of Asgard, Chris Bobbitt, ran the program through its paces. Spectators were impressed with the program's graphics capabilities (you can have up to 28 pictures of any size on a page at once), the ability to see the page on the screen before printing it, its ease-of-use (it's completely "what-you-see-is-what-you-get"), the multiple font features, and the line-drawing options.

Mr. Bobbitt did caution, "This isn't a full-scale desktop publisher, instead we like to call it a 'page-maker'", but most attendees felt the program would let the user do most anything that so-called desktop publishers could, and much more easily too. The general opinion from those who had a chance to use the program was that being able to see everything on the screen as it would appear on paper was a leap beyond anything else for the 99/4A or Geneve 9640.

When asked, attendees listed as reasons for purchasing the program many other things it offers including: the ability to use standard TI-Artist artwork, the ability to type in any direction, being able to print out the page at several dot densities, and also the ability to import TI-Writer files were all mentioned. Revievers also liked its Myarc HFDC and 9640 compatibility, and the general stability of the program (which users speculate was because of its lengthy testing period).

Continued on Page 2



PROGRAM FEATURES

- 28 Pictures/page
- Large font and a small font on each page
- 2 linestyles/page
- See page on screen before printing
- Print in single, double or quad-densities
- Type up/down/left or right
- Import TI-Writer files
- Save/load page
- Uses TI-Artist Instances & Fonts
- And much more!
- Requires Disk, 32K XB or E/A modules

$27.95

PLUS $0.75 S&H
Canadians add $0.50
Airmail add $2.75

ASGARD SOFTWARE
BOX 10306
ROCKVILLE, MD 20850
(703)255-3085
MASTERCARD/VISA

Features:

- Paste up to 28 pictures of any size on a page
- Type in one large font and one small font per page
- Two linestyles can be used at once
- Print out pages in single, double or "quad" density
- Type in any direction - up/down/left or right
- Includes utilities for converting TI-Artist fonts and instances, as well as creating two column text
- Import/Export TI-Writer files, save and load pages
- Geneve specific version included
- Use TI-Writer keys for editing text. Place pictures wherever the cursor is
- Requires 32K, Disk, Epson compatible printer.

Asgard Software
P.O. Box 10306
Rockville, MD 20850
(703)255-3085

Only $24.95

U.S. please add $0.75 S&H
Canadians please add $1.25 S&H
Europe/Asia/Australia please add $4.00 S&H

Comments

Amazing, a TI that runs on batteries

A three-part series on Jan Janowski's portable TI starts this month. His project, which has been in the works for about a year, was completed and demonstrated to the Chicago TI User Group in late spring. His machine consists of a black and silver console with an expansion memory and 512K RAMdisk mounted inside. He also has a parallel port to access a printer. Together with his custom-made 8-bank "super cart" and a battery he can take his TI to the beach, or anywhere, and do anything from word processing to spreadsheets without the need for an external power supply. Files that he writes he can save to the RAMdisk, which is battery backed so that when he turns off the TI the contents of the RAMdisk remain intact. Then, when he gets home, he can plug the TI into his Peripheral Expansion Box, download the files to a floppy drive for permanent storage.

Readers should not attempt to duplicate the portable TI project until the third and final installment in the series is published in the August edition. But it makes good reading for those who are interested in the possibilities that exist for the TI.

WE BLEW IT

Our apologies to CompuServe's TI-SIG. Laura and I were supposed to be the guests on a TI-Forum conference in June but we just forgot. We hope to have it rescheduled. Sorry.

MYARC Q&A TO RETURN NEXT MONTH

We haven't been running the Myarc Q&A column the past several months but we'll pick it up again starting in July. We were going to publish it this month but we ran out of space. Keep those questions coming.

—JK

READER TO READER

Lewis W. Shuck, 3457 Goldendale, Dallas, TX 75234, writes:

In an XBASIC program I am using multiple, one dimension, string variable arrays that I would like to operate on with one or more subroutines. This could be accomplished if there was a method for making an indirect reference to a string variable name, which I don't believe I can do in XBASIC.

I could use reader assistance in determining a way of accomplishing this function.

***Reader to Reader* is a column to put TI99/4A and Geneve 9640 users in contact with other users. Readers with a specific problem or question that may be answered by other readers is encouraged to submit an item. Send it to Reader to Reader, c/o MICROpendium, P.O. Box 1343, Round Rock, TX 78680.**

REVIEWED IN MICROPENDIUM

1984

February: B-1 Nuclear Bomber, Tandon TM-100 Disk Drive, Void, Beanstalk Adventure, Microsurgeon, On Gaming, Database 500.
March: Star Trek, Escape From Balthazar, Garkon's Getaway, Sky Diver, Mail-Call, Prowriter 8510 Printer.
April: Monthly Budget$ Master, Budget Master, Home Budget, Thief, Donkey Kong, Khe Sanh.
May: Companion Word Processor, Q*Bert, Mad-Dog I & II, Programs for the TI Home Computer.
June: Creative Expressions Accounts Receivable/Accounts Payable, CDC 9409 Disk Drive, Starship Concord, Lost Treasure of the Aztec, ASW Tactics II.
July: Theon Raiders, Introduction to Assembly Language for the TI Home Computer, Game of Wit, Pole Position
August: TE-1200, Tower, Galactic Battle, Galaxy
September: Wycove Forth, 99/4 Auto Spell-Check, QUICKCOPYer, Wizard's Dominion, Anchor Automation Mk XII Modem
October: Killer Caterpillar, ZORK I, Defender
November: 9900 Disk Controller Card/Manager, Super Bugger, Transtar 120S printer, Floppy-Copy, Data Base-X
December: Gravity Master, Data Base Manager System, Learning 99/4A Assembly Language Programming

1985

January: Super Sketch, Foundation Computing 128K Card, PTERM-99, TI-Runner
February: Super Extended BASIC, Beginning Assembly Language for the TI, ZORK II
March: Morning Star Software CP/M Card, WDS/100 Winchester Disk Drive, Sketch Mate, BMC Color Monitor
April: 9900 Micro Expansion System, Disk+Aid, Gemini 10X-15X
May: Character Sets and Graphics Design, Draw 'N Plot
June: GRAPHX, DATA BASE I
July: Acorn 99, Advanced Diagnostics
August: Model Dow-4 Gazelle, TI-Artist, PC-KEYS, Not-Polyoptics' Bankroll
September: Midnite Mason, Myarc 32K/128K Card, GRAPHX Companion
October: 4A/TALK, Extende BASIC II Plus, XB Detective, Console Writer 2.a
November: Foundation Z80A/80-column cards, 9900BASIC, Adventure Editor
December: Display Enhancement Package, Triple Tech

1986

January: BITMAC, Starcross
February: Night Mission, Peripheral Diagnostic Module, BA-Writer
March: Super Duper, Tunnels of Doom Editor, Business Graphs 99
April: U.S. Open Tennis, PRBASE
May: 4A Flyer, GRAM Kracker, Artist's Companion
June: Myarc Disk Controller Card, Maximem
July: Horizon RAMdisk, Old Dark Caves, Funlwriter, TI99/4A Macro Assembler
August: JOYPAINT 99, GPL Assembler, TI99/4A Intern, GPL Linker
September: Mechatronic 128K Card
October: TI-Forth Utilities, CorComp Memory Plus
November: Submarine Commander, PEP, MAX-RLE
December: GK Utility I and II and GRAM Packer, X-10 Powerhouse, RAVE 99/101.

1987

January: MG DISkASSEMBLER, Myarc XBII
February: TI-Tax, Mechatronic Mouse
March: Wycove Forth version 3.0, DIJIT Systems RGB Conversion Kit, Spad XIII Flight Simulator
April: Geneve 9640, Disk Utilities
May: QS-Solitaire, Geneve 9640 (Part 2), Technical Drive, Console Calc
June: Character Sets and Graphic Design III, Writerease Ver. 1.1, 4A DOS, Prescan_It
July: Junkman Junior, Avatex 1200/1200hc modem, Bubble Plane
August: Prostick, The Brain, Rocketman, Menu Ver. 6.3
September: TI-IBM Connection, Super Extended BASIC
October: Fontwriter, Mechatronic 80-Column Card, Star NP-10 printer
November: Legends, Music Preprocessor, QS-Wheel, Spin-to-Win
December: Remind Me, Certificate 99, Myart-Art and Myarc Mouse

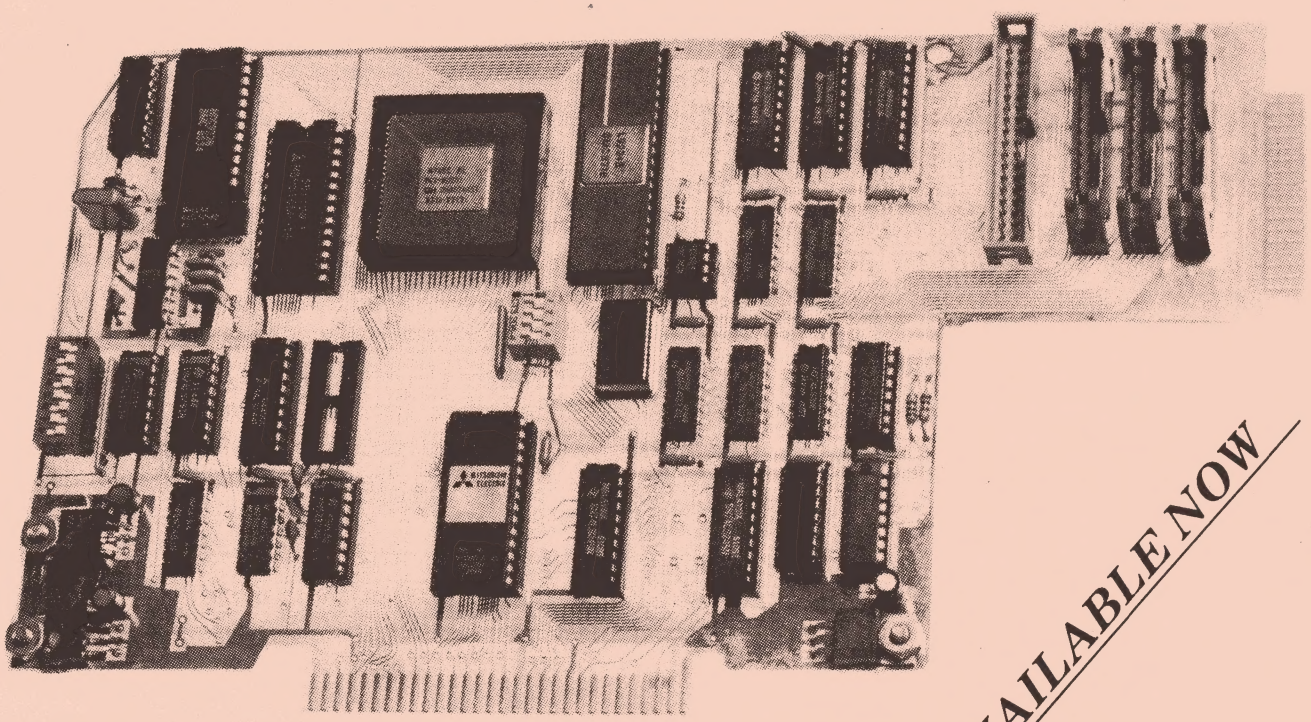
1988

January: Quik Font, EZ-Keys
February: Disk Utilities 4.0
March: Telco, String Master, Epson LX-800 printer
April: Super Space II, PC-Transfer, Calendar Maker, Archiver II
May: Plus!
June: Captain's Wheel 32K Memory Expansion, Desk Top Publisher Ver. 1.0, Texlink BBS
July: Artist Enlarger
August: Gramulator, Barrage
September: Myarc Hard & Floppy Disk Controller, Game Writers Pack I, Graphic Lister
October: Bunyard Hardware Manual, Writerease Update, M-Copy, Disk of Dinosaurs, Infocom Fast Loader
November:TI-Base, 3D-Maze, Macflix, Disk Labeler 99
December: P-GRAM Card, Epyx 500XJ Joystick, Enhanced Display Package, Starfleet Technical Drawings, Carfax Abbey, Rocketman

1989

January: First Base V1.0, Picture—It
February: Triad, Superbasic, P-Box Prototype Board, Keyboard Overlays, The Computer Phonebook, St. Valentine's Day Card, 1989 KBGB Girlie Calendar
March: NX-1000 Printer, Home Publishing on the 99/4A, Form-Shop, TELSUP V1.5, Boot/Menu programs, Arcade Action Software
April: Checkbook Manager III, TI-Runner Level Editor, TI-Writer V4.01, Artist Borders I, II, III, Multiplan Printer Codes
May: Jiffyflyer, Sector One Sector Editor, TI User Group Listing, Crypto, Giant Art Posters

THE MYARC Hard & Floppy Disk Controller with Streamer Tape Backup Support



AVAILABLE NOW

Features include:

- Hard drive transfer rate of 5Mbit per second, for speed comparable to an external RAM disk card
- Interfaces with standard, off the shelf, hard, floppy and streamer tape drives
- Built-in real time clock, for time and date stamping of files
- Supports up to four 5 ¼" and/or 3 ½" floppy drives, mix or match
- All disk formats, SS/SD (90K), DS/DD (320/360K) and DS/QD (640/720K) supported
- MYARC Disk Manager V, the most intuitive and user friendly manager available
- One year limited warranty, 12 months parts, 6 months labor, is standard, an optional two, three or four year extended warranty is available

MYARC, Inc., P.O. Box 140, Basking Ridge, New Jersey 07920, Phone (205) 854-5843

Feedback

Needs a program

I am not a programmer and am at the mercy of those who can and do program. I am a "user" of programs.

Sometimes a "general" database is just not right. It is too large, too intimidating, takes too long to set up and is not user friendly.

I need a program designed for making greeting cards only. It needs to be able to use TI-Artist and CSGD art (and others). It needs to be written so it uses all four quarters of the page: the face of the card, an instance on the inside cover, fonts for the verse page and a default (user-programmable) for the back page (a logo, so to speak). This program needs to be user friendly and even menu driven such as Great Lakes Certificate 99. (I use this a lot to make "cards.")

After the card maker is written for most dot matrix printers, I hope someone will be able to rewrite it for use with color printers such as Star's NX1000 Rainbow (which I hope to own someday).

I think others would like to have a program such as this and I hope someone will write it soon.

Laurel Crenshaw
Junction City, Oregon

Company not new

Re the MICROreview column (March 1989), Arcade Action Software is not new. Gene Hitz and his company have been supporting the 99/4A community for six or seven years now. But it is nice to see his programming expertise back in the limelight and it is neat Harry Brashear has given them the exposure they deserve.

Bill Gaskill
Grand Junction, Colorado

More on monitors

"Analog RGB Monitors," (MICROpendium, March 1989) was from our bulletin board (619) 278-8155, which we operate primarily for the promotion and support of our own products. However, we are willing to share information and downloads when source recognition is given, and encourage contributions from others.

"Analog RGB Monitors" was written a few years ago and has been updated once or twice. A significant deletion is Thomson CSF, who withdrew from the US market after a disastrous bout of trying to compete with Oriental sources for the IBM PC market.

We want to expand our list to include 80-column Analog RGB monitors conforming to PAL specifications, so we would appreciate hearing from our friends overseas as to what's available in their country.

Thomas F. Spillane
DIJIT Systems
San Diego, California

Our source for the article was another bulletin board, which did not note its source. However, we did credit Thomas Spillane as author.—Ed

Source for program

In reference to Jim Swedlow's comments about Turbo Copy, I am sure this fine fast whole disk copier is in the public domain. Guy Romano lists Turbo Copy as program H257 in his Free Access Library of public domain TI software. It was probably added to the library sometime in 1986. In order for any software to be included, Dr. Romano requires a written note from the author stating that the program is PD.

I too use Turbo Copy for whole disk copying. It works only with the TI controller, but it has built-in error checking not found in other fast whole disk copiers. Turbo Copy contains a bug that shows up if you use the program to initialize new double sided disks. Turbo Copy writes zeros in bytes >12 and >13 of sector zero. These bytes are where the number of sides and density are indicated and should read 0201 for DSSD disks rather than 0000. Such a Turbo Copy initialized DSDD disk works fine most of the time. However, if you try to copy the Turbo Copy initialized DSDD disk file by file with either Birdwell's Disk Utilities or DM1000, the copying program will get confused if you use an uninitialized target disk. Not finding 02 (meaning two sides) at byte >12 of the original disk, the copying program will initialize the target disk as single sided.

Why do you continue to list ancient shows in your listing of TI fairs? Why not just highlight the future shows!

Charles Good
Venedocia, Ohio

Some readers requested it. They feel knowing that, say, TICOFF was in April will help them plan for their next April. Write Romano at 116 Carl St., San Francisco, CA 94117 — Ed.

Definitive DOS

AARGH! I'd like to know the reason for so many letters in defense of Myarc. At this point, I think it is nonsense to defend Myarc.

Too much time lost, too many possible 9640 users lost, but also too many TI users lost!

It is time for Myarc to work night and day 24 hours per 24 hours to regain the lost time.

I do not say this because I feel myself superior to many others, or perfect, but because it is an impossible situation to have a machine for two years and not yet have a working and, above all, *definitive DOS.* Nobody will start a serious software project for the 9640 until the DOS is definitive. Myarc can continue to say that the present DOS is definitive but time and, above all, bugs say the contrary.

Daniel Marini
Milan, Italy

More on Myarc

Regarding the letters in response to one complaint (Alan C. Fox, March 1989) about Myarc's support — or lack thereof — I have no axe to grind either way because I don't own any Myarc equipment, so allow me a few remarks from the sideline.

The respondents were happy with the support they received from Myarc though — in most instances — it involved much time spent on the phone. Is this common or expected procedure nowadays if one makes a purchase? Would they feel the same if it involved another type of electronic equipment like a TV, VCR or stereo? Or a household appliance?

Should one be able to operate the equipment (by reading and following the instructions provided) or should it be necessary to enrich the phone companies to get it to work properly? Perhaps I am out of step with the times, but I still expect the former.

Lutz Winkler
San Diego, California

'classic' game modules

DBT DataBioTics

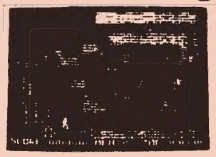
ALL THE NEW & EXCITING DataBioTics CARTRIDGES
FOR THE TI-99/4A COMPUTER
AT SUBSTANTIAL DISCOUNTS!


JUNKMAN JR.

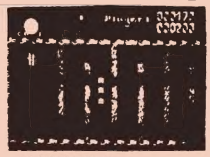
DataBionics is the largest developer and publisher of TI-99/4A Modules in the world! A California company made up of long time TI users who really know their stuff. They have taken the best commercial TI disk software and produced module versions in addition to introducing many new and outstanding products.


BARRAGE


MICRO-PINBALL

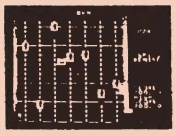

STAR RUNNER


TI TOAD


DRAGONFLYER


STAR TRAP


BLACKHOLE


SPY'S DEMISE


MICRO-TENNIS


MIDNIGHT MASON

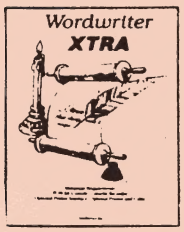

BURGER BUILDER


QMAZE


RED BARON FLIGHT SIMULATOR


BEYOND PARSEC

ENTERTAINMENT



BARRAGE.....16.95
SPOT SHOT.....16.95
BLACK HOLE.....16.95
SPY'S DEMISE.....16.95
JUMPY.....16.95
JUNKMAN JR. 16.95
STAR RUNNER.....16.95
DRAGON FLYER.....16.95
BEYOND PARSEC.....16.95
MICRO-PINBALL.....16.95
MIDNIGHT MASON.....16.95
BURGER-BUILDER.....16.95
TI-TOAD.....16.95
STAR-TRAP.....16.95
SPAD XIII-RED BARON FLIGHT SIMULATOR.....24.95
SORGAN II.....16.95
Q MAZE.....16.95
D*STATION.....16.95
MICRO-TENNIS.....16.95


STARGAZER 1/2/3

MANAGEMENT

INFORMATION MANAGEMENT MODULES

WORDWRITER XTRA (wordprocessor module).....34.95
WORDWRITER XTRA+ (with printer interface & cable54.95
SUPERSPACE 32K MemoryModule.with freemanuals....59.95
SUPERSPACE II (32K Memory Module).....59.95
SUPER FORTH (requires SUPERSPACE II).....17.95
DESK TOP PUBLISHER MODULE.....39.95
TI PLANNER MODULE (mini spread sheet).....24.95
PRO TYPER (touch typing tutor on module!).....19.95
STARGAZER I,II & III (astronomy course on module39.95

TEX★COMP

America's Number One TI computer retailer

P.O. Box 33084, Granada Hills, CA 91344

VISA & MASTERCARD HOLDERS CALL DIRECT

VISA MasterCard

(818) 366-6631

24 HOURS A DAY
7 Days a Week!

TERMS: All prices FOB Los Angeles. For fastest service use cashier's check, or money order. Add 3% shipping and handling ($3.00 minimum). East of Mississippi, 4% (free shipping on all software orders over $100.00). COD to be paid by cash or certified check. All TI products are sold with the original manufacturer's guarantee only (sent on request). Prices and availability subject to change without notice. We reserve the right to limit quantities.

NOTE: Payment in full must accompany all orders. Credit Card, Company Check or Money Order for immediate shipment. Personal checks require up to 4 weeks to clear. California orders add 6½% sales tax.

BASIC

Using speech on the TI

By REGENA

One reason the TI99/4A computer has been popular is that it has speech capabilities. In order to hear the speech on the computer, you need the Speech Synthesizer (a little accessory that plugs into the side of the computer console) and a module that uses speech.

The Scott Foresman module "Early Reading" requires the Speech Synthesizer to even work. Their later modules are examples of modules that use speech if the Speech Synthesizer is attached.

To use speech in your own programs, you need the Speech Synthsizer and the Speech Editor module, the TI Extended BASIC module, or the Terminal Emulator II module. The Speech Editor and TI Extended BASIC modules have a list of words that can be used, and you are pretty much limited to those words. You can make up a few more words by combining the existing words. The main commands are CALL SAY and CALL SPGET. See the manuals for the instructions for those modules to get an idea of how to use the speech and how to combine words.

I have always used the Terminal Emulator II module for speech because it is easier to program. Be sure you have the Terminal Emulator II (or 2) module. Terminal Emulator I cost about the same and only had the terminal capabilities. Terminal Emulator II is like two modules — it has the telecommunications features and the speech features. Rather than using a set list of words, you can ask the computer to say *any* text aloud.

First, plug in the Terminal Emulator II module and make sure the Speech Synthesizer is connected properly. When the title screen comes on, press any key. When the master selection list appears, choose 1 for TI BASIC. BASIC programming will not appear any different from any other time you are in BASIC programming, but with this module you can use speech. To use the speech, you will OPEN the speech file, then PRINT whenever you want the computer to say something. For example, start with this simple program.

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
200 CLOSE #1
210 END
```

Line 120 OPENs the OUTPUT file #1 (use any number) for speech. Line 200 CLOSEs the file when the program ends. Now to hear the computer say anything, use a PRINT #1 statement. For example, add this line then RUN the program.

130 PRINT #1:"DAD AND MOM"

Try some examples of your own between lines 120 and 200.

There are several inflection symbols. I like to use the carat "^ " before words to get a deeper voice. Try this line:

130 PRINT #1:"^ DAD AND MOM"

The ^ inflection symbol can change the inflection in a word and indicates a primary stress point in a sentence (you may use only one^ per line). You can hear the influence of this symbol in the following program.

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
130 PRINT #1:"^TRY THIS SENT
ENCE."
140 PRINT #1:"TRY ^THIS SENT
ENCE."
150 PRINT #1:"TRY THIS ^SENT
ENCE."
200 CLOSE #1
210 END
```

Other inflection symbols are the underline "_" and the greater-than symbol ">". "_" indicates a secondary stress point. ">" changes the stress point within a word. See if you can hear the difference when you put these two lines in your sample program:

130 PRINT #1:"^ ELECTRIC"

140 PRINT #1:"^ >ELECTRIC"

You may be able to hear the difference more with this sample:

130 PRINT #1:"^ WONDERFUL"

140 PRINT #1:"^ >WONDERFUL"

Try your own words with two or more syllables to see how this > symbol changes the sound.

Other characters also affect the sound. The comma followed by a space cause a .1 second pause. Other symbols causing a .45 second pause are the period, exclamation point, question mark, colon and semicolon. They also cause inflection contours in the speech. Try these lines.

```
130 PRINT #1:"^USE A LONG SE
NTENCE, WITH SYMBOLS."
140 PRINT #1:"^USE A LONG SE
NTENCE WITH SYMBOLS"
150 PRINT #1:"^USE A LONG SE
NTENCE; _WITH SYMBOLS!"
160 PRINT #1:"^USE A LONG SE
NTENCE, _WITH SYMBOLS?"
200 CLOSE #1
210 END
```

The computer will say numbers, alphabetic characters and some special symbols. For example, "%" will be heard as "percent."

You may use the PRINT # statements just as PRINT statements in programming. Here is an example using a FOR-NEXT loop and pronouncing numbers.

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
130 FOR N=1 TO 9
140 PRINT N
150 PRINT #1:N
160 NEXT N
200 CLOSE #1
210 END
```

The computer assigns the variable N different numbers in the FOR-NEXT loop. Line 140 prints the number on the screen, and Line 150 says the number.

The following program shows an example using the words to be spoken in DATA statements. A FOR-NEXT statement is used to READ the word from DATA then speak it. Since my personal preference is to use the^ symbol, Line 170 shows how I can add that symbol to every word — combining two strings.

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
130 PRINT #1:"^HERE ARE THE
CHILDREN."
140 FOR K=1 TO 8
150 READ C$
160 PRINT C$
170 PRINT #1:"^"&C$
180 NEXT K
```

(See Page 11)

REGENA ON BASIC—

(Continued from Page 10)

```
190 DATA CHERY,RICHARD,CINDY
,BOB,RANDY,BRETT LYNN,BRAD,K
ELBY
200 CLOSE #1
210 END
```

Notice when you RUN the program that some of the names don't sound like they should. With this method of programming speech, simply spell the words differently to get the correct sound. This next program example has two data items for each name spoken. The first item is the name C$ as it will appear on the screen. The second item is the name D$ as it will be spoken by the computer. Line 150 reads two items at a time, C$ and D$. Line 160 prints C$ on the screen. Line 170 combines the ^ symbol with D$ and then will say the name.

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
130 PRINT #1:"^HERE ARE THE
CHILDREN."
140 FOR K=1 TO 8
150 READ C$,D$
160 PRINT C$
170 PRINT #1:"^"&D$
180 NEXT K
190 DATA CHERY,SHERY,RICHARD
,RICHRD,CINDY,SINNDY,BOB,BOB
195 DATA RANDY,RANDY,BRETT L
YNN,BRET LIN,BRAD,BRAD,KELBY
,KELBY
200 CLOSE #1
210 END
```

As you program in TI BASIC and use speech with this module, you'll find you will need to experiment with spelling to get the computer to say what you want it to, but I find this method of programming relatively easy (and fun) to do.

Another option we have in using speech is specifyiing the pitch and slope. The pitch is the highness of the spoken sounds, and slope is the rate at which the pitch changes in the spoken phrase. To set the pitch and slope, use the form //xx yyy where xx is the pitch period and yyy is the slope level indication, and they are separated by a space. The pitch may be a value from 0 through 63. 0 is a whisper, 1 is a high pitched voice and 63 is a low pitched voice. The slope level may be a number from 0 to 255. The manual recommends the slope to be 3.2 times the pitch. There are also these limits:

yyy < [xx-1]*16 or yyy < [63-xx]*16

or the speech may be garbled.

The slope and pitch can take hours and hours of experimentation — but it can also be fun to hear the range of the computer. Here is a simble example of using pitch and slope. The computer will vary the pitch P from 0 to 63. The slope will be the recommended factor of 3.2 times the pitch. Line 150 combines the double slash with the pitch and slope for B$. Line 160 prints B$ on the screen, and Lines 170 and 180 say a phrase with the pitch and slope.

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
130 FOR P=0 TO 63
140 S=INT(P*3.2+.5)
150 B$="//"&STR$(P)&" "&STR$
(S)
160 PRINT B$
170 PRINT #1:B$
180 PRINT #1:"A B C"
190 NEXT P
200 CLOSE #1
210 END
```

With different pitches and slopes, you can make the computer sing. Remember, the slope does not have to be 3.2 times the pitch. However, to save time in experimentation, I used that factor for this sample:

```
100 REM USING SPEECH
110 REM TERMINAL EMULATOR II
120 OPEN #1:"SPEECH",OUTPUT
130 PRINT #1:"//43 128"
140 PRINT #1:"TAKE"
150 PRINT #1:"//24 77"
160 PRINT #1:"ME"
170 PRINT #1:"//28 90"
180 PRINT #1:"OUT"
190 PRINT #1:"//35 112"
200 PRINT #1:"TO"
210 PRINT #1:"//40 128"
220 PRINT #1:"THE"
230 PRINT #1:"//34 109"
240 PRINT #1:"BALL"
250 PRINT #1:"//43 138"
260 PRINT #1:"GAME"
270 CLOSE #1
280 END
```

Well, this example sings about as badly as I do, but you get the idea. This is a start, and you would experiment with different pitches, slopes and inflections to get the sounds better. Other programmers have really made some good singing programs.

The Terminal Emulator II module also allows another kind of speech programming, using allophone speech. Use the following OPEN statement:

OPEN #1:"ALPHON",INTERNAL

A list of allophones is given in the manual, and you can get exact sounds by using numbers rather than experimenting with spelling phonetically. For example, the first six allophone numbers correspond to six different pronunciations of the "a" sound.

I hope this article gets you going on using speech in your own programs. The main ideas are to use the Terminal Emulator II command module with the Speech Synthesizer and to experiment with different spelling (or allophones), pitch, slope, inflections, accents and symbols.

EXTENDED BASIC

Sending it to the dump

By JERRY STERN

That's right. Good weather's here. Time to get together all the old goodies and send them to the dump. That's a screen dump, by the way. Better get organized before the great outdoors gets too tempting. Perhaps adding a screen dump routine to your favorite programs would keep you inside. (Is that good or bad?) I know, this sounds too much like work for good weather. Not really. If the dump routine is called as a subprogram, it can be merged into any Extended BASIC program with only the addition of a well placed line for calling the subprogram.

On many programs, the design of the screen is so much better than the design of the printout, it would really be preferable to just use a copy of the screen itself as a reference of the program's results. For some programs, a character by character copy of the screen will be perfectly adequate. Other programs, particularly those which have used CALL CHAR to redefine some of the display characters, will require a screen dump that prints out the modified graphics as they were displayed.

The first case is easy. The command CALL GCHAR can be used to 'get' the 'char' of each position of the screen and print out the same ASCII character.

The second method is more difficult. The Texas Instruments method of storing character patterns in memory is based on the number 16. Actually, it is a number in base 16, and 16 digits long. Since each character fills a display area of 8 by 8 dots or pixels, each of the 16 characters identifies four of those dots as either 'on' or 'off.' The pattern starts at the top of the character, representing each horizontal line with two hex digits. So the left four rows of the character pattern are stored as the first, third, fifth, seventh... etc. digits, and the even digits keep the information on the right-hand side of the pattern.

So far, this has just been a review. The problem in printing out these patterns lies in converting these 16 character strings to a code that the printer can understand as a graphics mode command. Unfortunately, not all printers use the same command methods. The technique we'll use will work on the TI 99/4A Impact Printer. Because that printer is really an Epson MX 80 with a 2K print buffer and a serial card, this technique will also work on any printer that follows the nearly standard Epson codes.

The Epson printers use a number in base 256 to pass along the graphics information for special printing. The number will be as long in digits as the pattern is wide. Our 8 by 8 pattern will be represented as an 8-digit list of numbers between zero and 255. The order of the pattern is different from the TI method. Each vertical row is represented by one of those digits, and passes strictly left to right. These methods of storing information are dramatically different. The number base is different. The order of steps is different. The first character of a TI pattern, for example, is partially represented in the first four digits for the EPSON. The other way, the first EPSON code is a part of the graphics of 8 of the 16 TI characters. The conversion will have to be done as a test of each of the 64 dots in the pattern to check if a particular dot is 'on' or 'off.'

Different printers will use different type faces for their printouts. Some will use elite, some pica, some compressed draft mode, and others will simply print badly.... Despite the limits, DUMP is still useful.

Let's start with a test screen. It will make testing the dump routines easier. The test routine should be saved by itself on disk so that it can be loaded into memory as a 'fresh copy' of the routine is needed. The DUMP and DUMP3 subprograms should be saved on disk in their own files, in MERGE format. Merge them into SCREENTEST for testing, and into any other program in the same way. Look at line 90 of SCREENTEST. P$ is the name of the printer for your system. Change "PIO" to "RS232.BA=4800," or whatever you need. The test routine will pass this printer name along to the subprograms, so the change will only need to be made in this one spot.

```
120 CALL CHAR(42,"0704070707070700
0F063F2626060910FF117B7B7B71FF2020
20E00000000080")
```

Oops, that code is a bit more than 16 characters long. That statement wouldn't work in console BASIC, but Extended BASIC allows hex strings in the CHAR statement up to 64 digits long, for defining four consecutive character patterns. That change was intended for the purpose of making it easier to define the shapes of magnified sprites. When using magnification factors three and four, sprites consist of four consecutive characters arranged in a square pattern of upper left, then lower left, upper right, and lower right.

Line 120 of the test program defines new shapes for characters 42 through 45, or for *, +, "," and −. Let's do one more.

```
130 CALL CHAR(37,"10102828444482F
E")
```

Now that these characters have new shapes, they can be used to draw pictures. No, I'm not telling what it is You'll have to type it in. SCREENTEST provides a test screen to print out. The next step will be a dump program for printing it out.

The subprogram DUMP is a very basic program. A pair of nested loops CALLs the GCHAR command for each character of each row of the screen, and then prints out the accumulated patterns. Limitations: Characters that have been redefined with the CALL CHAR command will print out as the standard character normally associated with that ASCII character. Our characters of "*" and "+" have been changed to shapes more suited to a graphic display, but they will still print out as their original shapes, as defined by the pattern tables in the printer used for the dump. Dif-

(See Page 13)

EXTENDED BASIC—

(Continued from Page 12)

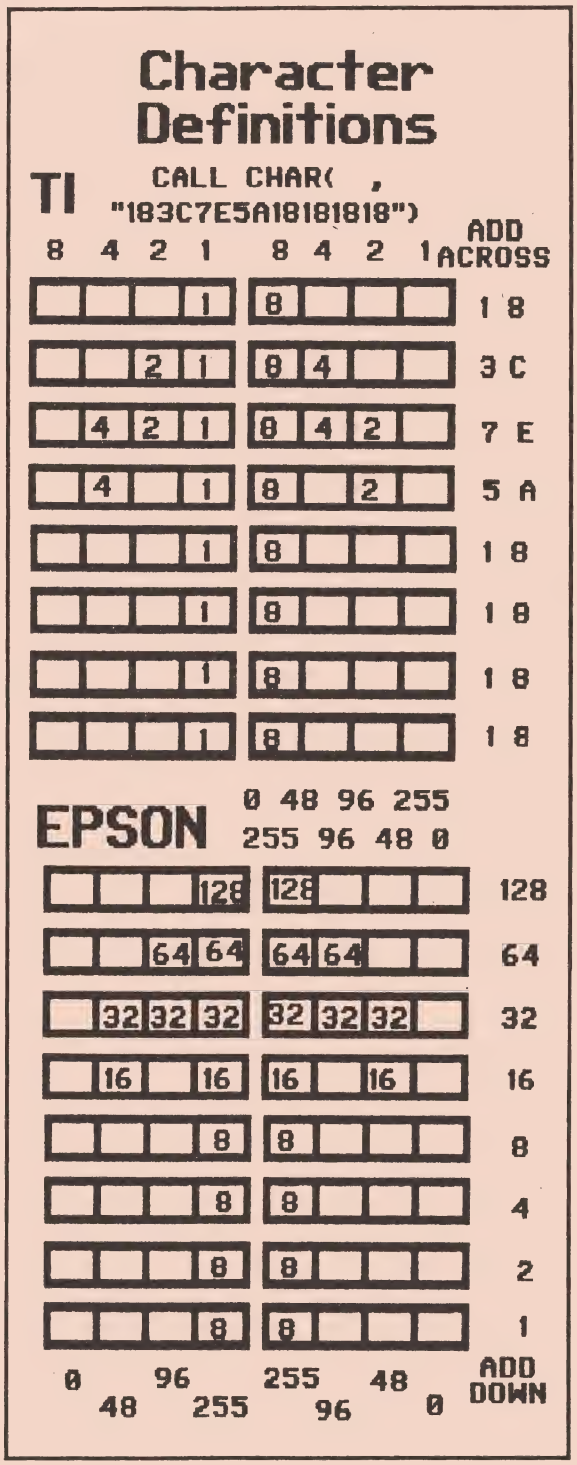
ferent printers will use different type faces for their printouts. Some will use elite, some pica, some compressed draft mode, and others will simply print badly. Sprites will not show up because they cannot be detected by GCHAR. Despite the limits, DUMP is still useful. Not all uses will require the printout of the redefined shapes, or of the shapes and locations of sprites.

In theory, the sprite positions could be identified with CALL LOCATE, the character locations converted to rows and columns, and the dot patterns superimposed on the patterns of the background characters. But the nature of sprite motion, along with the inherent speed limitations of Extended BASIC, and the problems of representing sixteen colors as shades of gray, make that kind of a total screen representation impractical. That could be a good project for programming in assembly language. The reasonable intermediate step between DUMP and this ultimate image capture system is DUMP3.

DUMP3 will print out all the patterns shown on the TI screen, except for sprites. Patterns will be shown as they are defined in memory, with no allowances made for the light/dark relationships that could have been reversed with a CALL to the TI command COLOR. The same simple loops of DUMP have been repeated, with a difference. Each character is checked for its current shape by the command CHARPAT. That command returns the hexadecimal code of each character's shape as a string variable.

So far, simple. There are two tricky parts yet to come. First, the process of converting hexadecimal TI character codes to Epson command codes will take time for each one. Each pattern should only be converted once, and only if needed for the screen being printed. Secondly, the conversion itself must be worked out.

The string array C$(143) will be used to store the converted character patterns. The DIMension is set at 143 because that is the maximum number that TI Extended BASIC can accept as an ASCII code. There are fewer codes than that, because the codes below 32 have no displayable shapes. A smaller array could be used by subtracting 32 from each ASCII character number, but the strings C$(0) to C$(31) will remain empty, and will not use extra memory because TI Extended BASIC does not allot memory space for string variables until they are filled. As DUMP3 scans the screen, line 29300 checks to find if the member of the array assigned to the character found has already been defined. If it has not been converted, DUMP3 sends the TI code to another subprogram, CHARPRT, for conversion. Otherwise, the graphics command to print that shape is sent to the printer.



I separated the conversion routine from the dump routine for two reasons. First, mixing the complex conversion code, with its nested loops, into DUMP3, with more nested loops, would have created a debugging nightmare. It is far easier to write these functions into separate routines and test them separately. That's one of the basic principles of "modular programming." By breaking the code up into function-based chunks, the chunks, or in this case subprograms, can be debugged separately. Secondly, using separate subprograms allows CHARPRT to be used by itself in other projects later on.

CHARPRT starts with some basic setup work. The TI hex code is the string C$. That string is converted to 16 numbers from 0 to 15, all stored in the array T(16). Next, the array C(8) is set to zero. But isn't any variable initially set at zero by the computer? Yes, but CHARPRT will be called as a subprogram more than once, and will keep the values left in it from each previous use. This is very convenient for some uses, but this time causes an extra step of resetting a variable array.

Here come those nested loops. First, L=1 TO 2 is used to represent the left and right sides of the TI character pattern, or the first, third, fifth, etc. characters on the left, and then the matching characters representing the right side. The loop FOR L2=L TO 16 STEP 2 steps through those characters, working down the

(See Page 14)

EXTENDED BASIC—

(Continued from Page 13)

TI pattern one side at a time. The final loop, FOR P=0 TO 3, where P stands for powers of two, steps through the four dots represented by each of the TI hexadecimal characters.

Finally, there it is. The one monster line of the program.

29360 IF (T(L2)AND 2^P)=2^P THEN C((L-1)* 4+4-P)=C((L-1)* 4+4-P)+2^(INT((16-L2)/2))

The loops set up by L (1 to 2), L2 (L to 16 step 2), and P (0 to 3) combine to form two times eight times four, or 64 cycles through line 29360. That's once for each dot in the character. Each time, the line checks if there is a dot in a particular location of the character, and then adds the correct power of two to C() to form the Epson code.

To make sense of that line, we'll relabel some bits of it. Think of INT((16-L2)/2) as the height of the dot being examined. 2^(INT((16-L2)/2) then must be the value that must be sent to the printer to print that particular dot. The sum of those powers of two for each dot is the code to print one vertical row of eight dots of the pattern.

(L-1)* 4+4-P is the position left to right of the dot being examined. That algebraic expression, enclosed in C(), is the code sum for that vertical line. T(L2) examines one digit of the TI code. 2^P is simply the number one, two, four or eight as P loops through zero to three. So, if T(12) is 12, and 2^P is eight, how much is 12 AND 8? Those of you who said "twenty!" keep reading. If you said eight, very good! AND does not mean "plus," but here is used as a binary AND command. Let's try it again:

Twelve=eight + four. (2^3 + 2^2)

Eight=eight. (2^2)

12 AND 8 equals the common powers of two, or 2^2.

Another example: How much is seven and eleven?

Eleven=eight + two + one. (2^3 + 2^1 + 2^0)

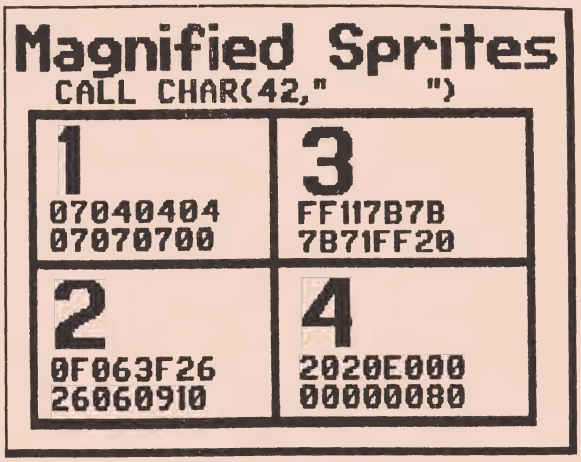
Seven=four + two + one. (2^2 + 2^1 + 2^0)

11 AND 7= 2^1 + 2^0=2 + 1= 3.

Line 29360 checks, then, for a dot in a particular part of a particular digit by ANDing that digit with the number 2 raised to the power of the dot being checked for. The binary AND isn't useful very often, but here it is the most elegant way to break down the number. The alternative coding would be about four lines of IF statements, each checking if T(L2) is equal to 4, 12, 13, 14 or 15, then the "4" dot, or 2^2 dot is "on." Too messy and slow for me.

Once all 64 dots have been checked, the accumulated values in C() are placed together in a string variable, right after the Epson command to print a single density graphic character eight rows wide: 27, 75, 8, 0. The combined string is sent back to the calling program, or in this case the calling subprogram, ready for printing.

Did you notice that in DUMP3, if you run the routine more than once, it must run faster each time? Because the variable values are not lost when a subprogram is exited, DUMP3 does not need to convert any character more than once, even on separate dumps at different times in program execution. The first dump will be slow. The second dump will keep ahead of the printer easily, and keep a printer buffer active as well.




```
100 ! SCREENTEST: TESTER FOR
 SCREEN DUMPS !100
110 CALL CLEAR !209
120 CALL CHAR(42,"0704070707
0707000F063F2626060910FF117B
7B7B71FF202020E0000000000080")
!049
130 CALL CHAR(37,"1010282844
4482FE")!013
140 PRINT " *, *, *,":" +- +
- +-": :" ON STRIKE": :" FOR
 BETTER PRINTOUTS!": : : : :
 : :!229
150 CALL HCHAR(23,1,37,96)!2
36
160 CALL VCHAR(2,31,37,21)!2
38
170 CALL VCHAR(2,2,37,21)!18
7
180 DISPLAY AT(24,1):"PRESS
1(TEXT) OR 2(GRAPHICS)" !157
190 CALL KEY(0,K,S):: IF S<1
 THEN 190 !004
200 IF K=49 THEN CALL DUMP(P
$)!244
210 IF K=50 THEN CALL DUMP3(
P$)!032
220 STOP !152
```

```
31195 SUB DUMP(P$):: !SCREEN
 DUMP !159
31200 OPEN #9:P$ !199
31205 FOR R=1 TO 24 :: A$=""
 :: FOR C=1 TO 32 :: CALL GC
HAR(R,C,X)!221
31210 A$=A$&CHR$(X)!219
31215 NEXT C :: PRINT #9:A$
:: NEXT R !122
31220 CLOSE #9 :: SUBEND !20
1
```

```
29270 SUB DUMP3(P$)!184
29275 ! SCREEN DUMP BY ACTUA
L CHARACTER PATTERN JLS 6/89
 !056
29280 DIM C$(143)!193
29285 OPEN #8:P$&".CR",OUTPU
T :: PRINT #8:CHR$(27);CHR$(
65);CHR$(8);!212
29290 FOR R=1 TO 24 :: FOR C
=1 TO 32 :: CALL GCHAR(R,C,T
)!109
29295 IF T<32 THEN T=32 !183
29300 IF C$(T)="" THEN CALL
CHARPAT(T,S$):: CALL CHARPRT
(S$,C$(T))!252
29305 PRINT #8:C$(T);!043
29310 NEXT C :: PRINT #8:CHR
$(10);CHR$(13):: NEXT R :: C
LOSE #8 !199
29315 SUBEND !168
```

```
29320 SUB CHARPRT(C$,T$)!131
```

(See Page 42)

LOADERS, MODULAR PROGRAMMING, LINKAGES & OVERLAYS

Overlays with E/A and BASIC

By MERLE VOGT

This is the fourth of a five-part series.—Ed

We are finally getting into some real meat. Previously I mentioned some very large programs, such as Legends and Old Dark Caves, that are 4 or 5 times larger than can be fitted into 99/4A RAM.

What we do to use large programs is create a series of modules in assembly language. Each is a subtask, perhaps in a game it would be one monster. Then, as the job progresses, we load and run these modules. In the easiest scheme we could load modules sequentially and plow through the program step by step. But it is not absolutely required that we do it that way. More on this later.

It is clear that we cannot just blindly use a long chain of CALL LOAD commands to make a run. Memory and the REF/DEF table gets filled up very fast. So we have to play some sneaky tricks on the *blind side* of the loader.

The key is found in some data elements used by the loader which I have not yet emphasized. First, let us examine the Editor/Assembler and Mini-Memory systems. Both use an area of RAM named the UTLTAB (Utility Table), which is a misleading name since its connection with the utilities previously discussed is virtually non-existent. But we are stuck with it.

Specifically we are interested in some addresses in UTLTAB. These are named FSTHI, LSTHI, FSTLOW and LSTLOW.

FSTHI means FirST free address, HIgh memory
LSTHI means LaST free address, HIgh memory
FSTLOW means FirST free address, LOW memory
LSTLOW means LaST free address, LOW memory

Below I have tabulated these items, their addresses and their contents, for the Editor/Assembler and Mini-Memory. The function of each item is similar but the addresses and contents may differ slightly.

Data Item	E/A Location	E/A Value	MMM Location	MMM Value
FSTHI	>2024	>A000	>7022	>A000
LSTHI	>2026	>FFD7	>7024	>FFE0
FSTLOW	>2028	>2676	>7026	>2000
LSTLOW	>202A	>3F38	>7028	>3FFF
FSTMEDIUM			>701C	>7118
LSTMEDIUM			>701E	>8000

Look in the Mini-Memory manual, page 74, and the E/A manual, page 265.

The values shown are placed into the locations by the Load and Run phase or the CALL INIT phase of operations.

Warning: This does not apply to the Extended BASIC CALL INIT. It does a different setup in most aspects. This will be discussed next month.

Now, follow this carefully. When you execute the loader it looks at the length of modules and updates addresses accordingly. Consider the Editor/Assembler. You bring up menu option 3 (Load and Run). You type in an object module name, DSK1.ENTER1OBJT, hit Enter and the loader runs. It looks in FSTHI and sees the value >A000, so it starts placing the code of ENTER1 into that space in RAM. It finds the length of ENTER1 and increments FSTHI accordingly. Just for some numbers assume that ENTER1 is >1100 bytes long. FSTHI becomes >B100. Also, there must be one, or more, DEFs in ENTER1. The loader looks in LSTLOW (=>3F38) and plugs in three DEFs at addresses >3F30, >3F28 and >3F20, then updates LSTLOW to be >3F20. (Refer back to the modules in Part 1 of this series.) Then the loader asks for more input.

Now we type DSK1.ENTER2OBJT, and hit Enter. Module ENTER2 will be loaded into RAM starting at location >B100. Assume it to be >0700 bytes long. FSTHI is updated to equal >B800. ENTER2 had two DEFs. They will go into locations >3F18 and >3F10, and LSTLOW becomes >3F10.

Then we type DSK1.MAINOBJT, and hit Enter. MAIN is loaded at >B800. If MAIN was >1200 bytes long, then FSTHI becomes >CA00. There is one DEF, so it goes into >3F08 and LSTLOW becomes >3F08.

If we continued loading we will push on up in high RAM. Someplace we will get a module that won't fit. Then the loader goes to FSTLOW, which should still be >2676. If the modules will fit between >2676 and LSTLOW it is loaded and some more DEFs added to the REF/DEF table. If you run out of space then, you have built a program that cannot be run.

Reminder: All the above is *static* procedure, using the E/A. You can operate similarly using Mini-Memory. The REF/DEF table will be at locations >7FF8, >7FF0, >7FE8, >7FE0, >7FD8 and downwards. The module locations at >A000, >B100 and >B800 do not change.

Okay, get a cup of coffee. I have gone through this mess in exhaustive detail because, once you get an iron grip on it, the rest is all downhill.

We will proceed on an overlay with these premises:

1. Use Mini-Memory environment.
2. Use a master control program in BASIC (not XBASIC).
3. Execute one assembly module at a time.
4. Make a three-module system.
5. Have modules No. 1 call in module No. 3, which will call module No. 2, which will terminate the job.

We will make the loader put each new module into the RAM space occupied by the previous one, hence *overlay*. It is easy to *trick* the loader into doing this.

Consider FSTHI. It contained the value >A000. So, before we load a module, we place >A000 into FSTHI.

Warning: Proceed cautiously. The loaders are different, the Mini-Memory loader operates with a split REF/DEF table and places the DEFs into the RAM part of the REF/DEF table starting with the address found in LSTMEDIUM, at >701E, which contains >8000. So, in the Mini-Memory environment, you find

(See Page 16)

LOADERS, MODULAR PROGRAMMING, LINKAGES & OVERLAYS

(Continued from Page 15)

your DEFs at >7FF8, >7FF0, >7FE8 and downwards.

Therefore, to run an overlay you must reset LSTMEDIUM to contain the value >8000, erase all of the DEFs at >7FF8 and below, and reset FSTHI back to >A000.

Look at subroutine 6000 in the BASIC program. See Fig. 1, BASIC master control program.

Fig. 1
BASIC MASTER CONTROL PROGRAM

```
100 REM RUNNING MULTIPLE OVE
RLAYS. MINI-MEMORY WITH BASI
C     !219
110 CALL INIT !157
120 NEXTRTN=1 !235
130 REM !186
150 ON NEXTRTN GOTO 1000,200
0,3000,4000 !197
160 REM !186
1000 REM RESTORE THE POINTER
S     !243
1010 GOSUB 6000 !215
1020 REM PULL IN ROUTINE AAA
AMM   !242
1030 CALL LOAD("DSK1.AAAAMMO
BJT")!218
1040 REM NOW RUN IT !160
1050 CALL LINK("AAAAMM")!111
1060 NEXTRTN=3 !237
1070 GOTO 150 !229
2000 REM RESTORE THE POINTER
S     !243
2010 GOSUB 6000 !215
2020 REM !186
2030 CALL LOAD("DSK1.BBBBMMO
BJT")!222
2040 REM !186
2050 CALL LINK("BBBBMM")!115
2060 NEXTRTN=4 !238
2070 GOTO 150 !229
3000 REM RESTORE THE POINTER
S     !243
3010 GOSUB 6000 !215
3020 REM !186
3030 CALL LOAD("DSK1.CCCCMMO
BJT")!226
3040 REM !186
3050 CALL LINK("CCCCMM")!119
3060 NEXTRTN=2 !236
3070 GOTO 150 !229
```

(See Page 17)

Fig. 2 SUBROUTINE AAAAMM

```
 1                DEF    AAAAMM
 2                REF    WMBW,GPLLNK
 3      AAAAMM    BLWP   @AA2
 4                B      *R11
 5      MA        TEXT   'MESSAGE FROM AAAAMM'
 6      M1        TEXT   'PASSING CONTROL TO CCCCMM'
 7      OFSET     DATA   >6060
 8      STAT      EQU    >837C
 9      WS        BSS    32
10      AA2       DATA   WS
11                DATA   AA4

12      AA4       LI     R3,MA
13                LI     R4,46
14      AA5       A      @OFSET,*R3+
15                DECT   R4
16                JNE    AA5

17                LI     R1,MA
18                LI     R2,20
19                BL     *ALINE
20                LI     R1,M1
21                LI     R2,26
22                BL     @ALINE
23                LI     R6,10
24      TA        SETO   R7
25      TB        DEC    R7
26                JNE    TB
27                DEC    R6
28                JNE    TA

29                CLR    @STAT
30                RTWP

31      ALINE     LI     R0,>200
32                BLWP   @VMBW
33                BLWP   @GPLLNK
34                DATA   >4D00
35                B      *R11
36                END
```

COMMENTS

Line No.	Explanation
1	Defines (DEF) name AAAAMM
2	REFs to subroutines to be used in AAAAMM
3-4	Start execution of AAAAMM
5-9	Specify text and data items used
10-11	Specify the BLWP data items needed by line 3
12-16	Since we are running the the BASIC environment, we must add the offset value (>60) to each byte of text items to make them displayable
17-19	Display message "MA" using subroutine "ALINE"
21-23	Display message "M1"
23-28	Time delay to read the messages
29-30	Clears status and exit
31-35	Subroutine ALINE displays messages on screen and scrolls
35	End

LOADERS, MODULAR PROGRAMMING, LINKAGES & OVERLAYS

(Continued from Page 16)

```
4ØØØ REM END OF JOB !Ø97
4Ø1Ø DISPLAY "END OF OVERLAY
 JOB": : :!Ø99
4Ø2Ø STOP !152
6ØØØ REM RESTORE LOAD POINTE
RS, STARTING AT >7Ø22 !1Ø4
6Ø1Ø FIRSTHI=7*4Ø96+2*16+2 !
165
6Ø2Ø REM >AØØØ=16Ø,Ø !Ø25
6Ø3Ø CALL LOAD(FIRSTHI,16Ø,Ø
)!2Ø8
6Ø4Ø REM NOW ADDRESS >7Ø1E !
Ø47
6Ø5Ø LASTMED=7*4Ø96+16+14 !Ø
12
6Ø6Ø REM NOW NEED >8ØØØ !Ø48
6Ø7Ø G=8*16 !25Ø
6Ø8Ø H=Ø !255
6Ø9Ø CALL LOAD(LASTMED,G,H)!
245
61ØØ REM NOW CLEAR 3 ENTRIES
 FROM REF/DEF TABLE !185
611Ø DEFF=7*4Ø96+15*256+15*1
6+8 !248
612Ø FOR LOP=1 TO 3 !22Ø
613Ø CALL LOAD(DEFF,Ø,Ø,Ø,Ø,
Ø,Ø,Ø,Ø)!1Ø7
614Ø DEFF=DEFF-8 !171
615Ø NEXT LOP !129
616Ø DISPLAY "READY TO LOAD
NEXT MODULE": : :!Ø94
617Ø RETURN !136
618Ø END !139
```

That is all there is to it. Since you reset all the addresses, each time you call the loader it cannot tell that you have previously called it and it loads the module at location >A000 and builds the REF/DEF table from >8000 downwards. Thus you can chain three modules or a hundred sequentially into execution and never run out of RAM space. My example is very simple and would be rather impractical because it uses small modules and there would be excessive disk operations with a lot of such code. The normal technique would use large modules which would fill most of high RAM, >A000 through >FF00; one each CALL LOAD and reduce the number of disk accesses.

See Fig. 2, assembly routine AAAAMM.

Module BBBBMM will be similar to AAAAMM. Only the changes are shown in Fig. 3.

Fig. 3 BBBBMM CHANGES

```
B1                DEF   BBBBMM
B3    BBBBMM      BLWP  @BB2
B5    MB          TEXT  'MESSAGE FROM BBBBMM'
B6    M2          TEXT  'PASSING TO END OF JOB'
B1Ø   BB2         DATA  WS
B11               DATA  BB4
B12   BB4         LI    R3,MB
B14   BB5         A     @OFSET,*R3+
B16               JNE   BB5
B17               LI    R1,MB
B22               LI    R1,M2
```

Since each module is separately assembled, there is no need to change the symbol names OFSET, WS, TA, TB or ALINE. Even further, if you will not allow it to confuse you, it is not necessary to change the symbol names in lines B5, B6, B10, B11, B12, B14, B16, B17 or B20. Line B1 must DEFine the name "BBBBMM", and line B3 must use that symbol. Also, the text in lines B5 and B6 must be changed to reflect the new module.

It follows that in module CCCCMM line C1 becomes DEF CCCCMM and line C3 becomes "CCCCMM BLWP @AA2". Thus, in this example, after you have coded module AAAAMM and saved it you can get modules BBBBMM and CCCCMM by changing lines 1, 3, 5 and 6 and making two more saves to disk. Note that this is an abnormal example, just to get your feet in the door quickly. In the real world, overlay modules would all be completely different in function and length.

Here is a summary of the procedural steps to put all this together.

- Enter BASIC and code the master control program (Fig. 1).
- Call up the Editor/Assembler editor and code the module AAAAMM (Fig. 2), and save it as AAAAMMSRCE.
- Change lines 1, 3, 5 and 6 of the source to create module BBBBMM and save it as BBBBMMSRCE.
- Change lines 1, 3, 5 and 6 again to create module CCCCMM and save it as CCCCMMSRCE.
- Call up the assembler and run it three times, to get object modules named:

AAAAMMOBJT
BBBBMMOBJT
CCCCMMOBJT

- Enable the Mini-Memory cartridge.
- Load and run the BASIC program.

The results should be:

A. A message from routine 6000, "Ready to load next module."

B. Messages from module AAAAMM and a time delay to read them.

C. The message from routine 6000, again.

D. The messages from module CCCCMM, and the time delay.

E. The message from routine 6000, again.

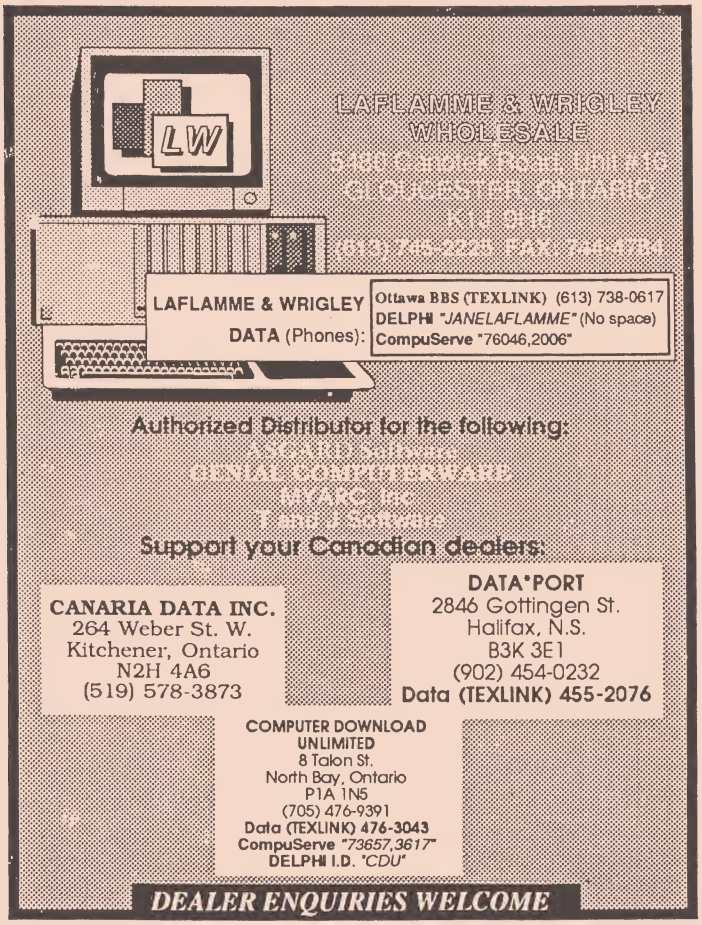
F. The messages from module BBBBMM, and the time delay.

G. The end of job message.

Now, review the whole package.

- The three modules are assembled in *relocatable* mode.
- The address pointer FSTHI is reset to the value >A000 to trick the loader into "forgetting" that it has previously loaded a module at that location. (See routine 6000, lines 6010-6030.)
- Similarly, LSTMEDIUM is reset to the value >8000 to make the loader place the REF/DEF table at the original location. (See lines 6050-6090.)
- Then the old entries are cleared out of the table area. In routine 6000 I show how to erase three DEF entries. (See lines 6100-6150.) In this example system there will be only one DEF made on each load. In actual systems you may have any number needed and must clear the REF/DEF table accordingly.
- From this point the potential is endless. As an example, you could create a module

(See Page 18)



LOADERS, LINKAGES & OVERLAYS

(Continued from Page 17)

at absolute origin >2000 in low RAM, and leave it there. (AORG >2000, see the chart at the beginning of this article.) Then you may call it as needed from any number of relocatable modules in high RAM. Note that although most addresses in a relocatable module are relocatable, this does not inhibit calling into subroutines with absolute origin addresses. Total organization of the system is required.

• The loader does not care which type of module that is to be loaded. It merely follows orders. You must completely control the steps executed. You can use module "XXXXXX," then overlay it, then bring module XXXXXX back in at any later step anytime it is needed.

This is how the large games operate. The same module may be loaded 100 times. Note: that was poorly stated. Properly, you should structure the program to keep that sort of module in RAM, not do 100 disk accesses. The load steps should be used to pull in modules of infrequent usage. Remember that you have RAM areas >A000 through >FFE0 and >2000 through >3FFF, which is more than 31K to work in. Also, since this is the Mini-Memory environment, there is available in RAM bank >7000 the area from >7118 through about >7F00. You can place about 3.3K of code there using the AORG scheme. Be careful not to overrun the REF/DEF table space. Thus, you can load and leave a fairly large amount of code that gets heavy usage and do the overlays as needed in the remaining space. Repeat: You must keep an iron grip on the system. Exhaustive flowcharts and documents are an absolute. It is not easy but you can make the 99/4A do anything you want to do. Just look in the software catalogs.

Warning: I have allowed another big trap to be placed in the modules AAAAMM, BBBBMM and CCCCMM. An attribute of modules, which hasn't been discussed in the literature, is *reusability.*

By that I mean that the modules are coded to be *one pass* and will bomb out if you try to run through them more than once. There are several aspects to be considered.

• If the module is reloaded, then it is a fresh copy of the code and reusability is not a factor.

• But, if the module is loaded once, and then executed several times, it must be reusable.

Reusable defined. Execution of a module can change various data items that were established by the load operation. To make a module reusable, the code must restore all changed items before the pass is completed. The items involved may be pointers, constants, counters, accumulators, text, etc. depending on the logic of the code.

The following instructions outline how to make reusable versions of modules AAAAMM, BBBBMM and CCCCMM.

A. Lines 12-16 are the area where these modules will bomb out if we reuse them. We cannot add the offset value (>60) to the message text more than once.

B. To make the modules reusable a *first pass through switch* must be added to the code. This consists of instructions that detect the *first pass through* and set a logic branch ("SWITCH"), which

(See Page 19)

LOADERS & OVERLAYS

(Continued from Page 18)

prevents executing lines 12-16 more than once.

C. Look at this added and changed code. Lines numbers 11B, 11C mean that you insert the lines following the original line number 11.

```
11B   SWITCH  DATA   0
11C   TESTSW  DATA   0
11D   AA4     C      @SWITCH,@TESTSW
11E           JNE    AA6
11F           INC    @SWITCH

12            LI     R3,MA

16B   AA6     NOP
```

At the start of the first pass through, the SWITCH equals zero. Lines 11D and 11E test the SWITCH. It is still equal to TESTSW so the logic falls into line 11F. That line adds 1 to SWITCH, closing the logic path. But lines 12-16 get executed this pass. On later passes the test causes a jump directly to line 16B, leaping over the code of lines 12-16.

> Many other elements of the code can get involved. There may be a counter which must be zeroed on the first pass and not thereafter.

Many other elements of the code can get involved. There may be a counter, or an accumulator, which must be zeroed on the first pass and not thereafter. A sticky case is where a file must be opened on the first pass, then closed on the last pass, which is some unknown number of passes after the first pass. Whooeee! Two switches would be needed.

It is difficult to give a lot of examples. The functions of the module are involved, and this dictates the type of added logic instructions required to make a module reusable.

Every item of data that got changed during the pass through must be restored to the original status or condition as the module is making its exit phase.

The final installment of this series next month will consider the use of overlays with Extended BASIC.—Ed

TEX-COMP
Celebrating Our Tenth Year

FREEWARE••

•• Public Domain and Shareware for the Texas Instruments TI-99/4A Computer.

AUTHORIZED SALES

ONLY **$4.95** Per Disk

Public Domain and Shareware Programs to Meet Your Every Computing Need.

BONUS

FREE DELUXE DISK STORAGE CASE WITH EACH ORDER OF FOUR OR MORE DISKS!!!

#84. GALACTIC BATTLE/SPY ADVENTURE
A pair of great commercial quality games from EB Software of TI Runner fame. Galactic Battle is a space "trek" type strategy game for one or more players. Spy Adventure is an adventure game that will keep you guessing for hours.

#85. AUTOBOOT UTILITY
This utility which can be installed on a disk loads and runs or displays most files. Now you can have a disk with exbasic programs, Editor Assembler programs and TI Writer files and run or display them all from exbasic.

#86. COLUMN TEXT III V3.2
A very useful utility for printing TI Writer and 99 Writer II files in separate spaced columns. Saves hours in producing a newsletter. Complete with documentation.

#87. ARCHIVER III
This utility allows you to "pack" or combine several files into one for space utilization. A number of boards are sending files packed to save transmission costs. This utility will let you pack and/or unpack these files.

#88. AUSSIE GAMES VOL 1
A collection of games from our friends down under. Includes a great card game and board game. Hours of fun and entertainment. Includes Matchmaker & TILO.

#89. PROCALC
This is an on screen calculator for decimal/hexidecimal conversions and much more. A must for the serious programmer.

 # 90. JET CHECKBOOK MANAGER
This checkbook manager is considered the ultimate with every feature you can think of for keeping track of your checking account and keeping records of your spending for budget and tax purposes. Complete with documentation.

#91. "THE MAZE OF GROG"(St. Valentine)
Ray Kazmer has created a great maze game with fantastic graphics and the characters from his now legendary "Woodstock" disk. Fun for all!!!

#92. HOUSEHOLD INVENTORY
Written by 99/4 programming great Charles Ehninger, this prize winner originally sold for $59.95. Keeps track of household, business or personal items by category and provides automatic updating for inflation etc. A must for tax and insurance records!

#93. THE 1989 KBGB GIRLIE CALENDAR
This latest offering from programming master Ken Gilliland prints out a jumbo 12 month calendar with a knock-out centerfold pinup for each month. If you like our #14 Figure Study disk, you will flip over this one. For Adults Only!! Exbasic & d/m printer.

#94. GREAT 99/4A GAMES VOL. III
If you have seen vols. 1 & 2 of this series you know we only provide the very best. This latest volumn is also filled with a collection of great ones!

#95. WEATHER FORECASTER
The weather predictions are amazingly reliable and accurate! A great game "Lawnmower" and a mini database are also included to make this disk a fantastic value.

#96. STATISTICS & SORTING
Two great assembly utilities by John Clulow. STAT is a set of statistic routines for use in exbasic. SORT allows sorting by two separate fields and a choice of two types of sorts.

#97. MEMORY MANIPULATOR
This powerful utility lets you explore the entire memory in your 99/4A system and take apart what you find. User friendly!

#98. DAYS OF EDEN & DOORS OF EDEN
Two bible games)non-fiction) that work with the TI Adventure Module.

#99. GREAT 99/4A GAMES VOL. IV
This disk features the works of J. Peter Hoddie. All of these games are of commercial qualaity and well worth the donation requested!

#100. ASSULT THE CITY (T. of DOOM)
An exciting game for use with the Tunnels of Doom module. Several Exbasic bonus games are included.

#101. ENCHANCED DISPLAY PACKAGE
This screen enhancement utility lets you do 40 columns, windowing, reverse scrolling, clock/alarm, and a whole host of other great tricks in exbasic. Fully documented.

#102. COLOSSAL CAVES ADVENTURE
This classic adventure now available for the 99/4A is what led to the Zork series. Hours of text adventuring.

#103. SORGAN, THE 99/4A ORGAN
This program which is currently selling for big bucks on module turns your 99/4A into an electronic organ. Sound effects, different instruments and voices, chord forms, color graphics with complete control of all.

#104. C99 COMPILER AND LIBRARY
This two-sided (flippy) disk gets you into C programming with your 99/4A. Comes with a great collection of utilities such as text & graphics. (E/A)

#105. KING'S CASTLE+
A great arcade style assembly game formerly offered on module. Also includes an EB "Trek" game and a collection of sprite & graphics from Tigercub's Jim Peterson.

#106. QUEST (Dungeons & Dragons)
One of the best D&D games around! You must destroy the Dark Lord to free your homeland! Complete with documentation on disk.

#107. STAR TREK MUSIC ALBUM
Ken Gilliand's music and graphics version of the TV theme and the three motion pictures. (Exbasic)

#108. FUNLPLUS BY JACK SUGHRUE
Fantastic disk packed with Funnelweb (#42) templates, utilities and prog. to augment and configure Funnelweb. Unbeliveable collection of fantastic aids to make the best even better!

#109. TI-WRITER MINI MANUAL
This disk prints out a five page TI Writer manual with everything you need to know to use TI Writer or the many clones such as 99Writer II. Additional aids for using this powerful word processor are included.

#110. DISK + AID
A powerful disk sector editor formerly sold for $20. Menu Driven and easy to use.

#111. POP MUSIC & GRAPHICS
This exciting disk from Germany features music/graphics written in 100% assembly and what comes from the TI sound chip is sure to astound you.

#112. INVOICE PACK
An excellent invoice preparation and printing program with instructions on how to modify it for your own business.

#113. LABEL MAKER 3
A collection of label programs to create mailing and disk envelopes, disk labels and much more!

#114. PANORAMA
A drawing and illustration program that compliments Graphx and TI Artist. A must for the serious 99/4A artist!

#115. GRAPHICS DESIGN SYSTEM
A complete system for creating graphic screens in full color for your programs by J. Peter Hoddie. Fully documented.

#116. FOURTH TUTORIAL
A lesson in FORTH programming on how to create graphics.

#117. UNIVERSAL DISASSEMBLER
This powerful utility written in Forth allows disassembly of programs off disk in any format, in memory, and even off of P-Box cards. Very complete with some very unique features.

#118. FAST TERM
One of the most popular and recommended of the 99/4A terminal emulator programs. Supports TE-II, ASC11, and X-Modem transfers, print spooling and more. Loads from Exbasic or E/A.

#119. RAG LINKER
A utility for converting DIS/FIX 80 assembly object code files to PROGRAM image. This allows files to load faster and take up less space on disk. Full Doc

#120. BITMAC
The original BITMAC is now available at $4.95 with all original documentation. A powerful graphics program for the 4A which lets you print where you want..even over pre-existing text. Create great graphics in 16 colors, print text sideways, mirror image, upside down etc. etc. A **must** for anyone into 99/4A graphics. Comes with second bonus disk with utilities such as sign & banner makers. Even can computer generate your own signature!

#121. SUPER YAHTZEE & WHEEL II
If you like Yahtzee this disk is for you. A great version written in high speed assembly. Also included is another version of Wheel of Fortune which also lets you create your own puzzles with a puzzle edit program included.

#122. ADULT ADVENTURE
A trily adult adventure for use with the TI Adventure Module. Also included is a bonus adventure (not adult) "LOST GOLD" which is one of the better ones we have seen recently.

Send order and make checks payable to **TEX-COMP**

P.O. BOX 33084 — GRANADA HILLS, CA 91344

TERMS: All prices F.O.B. Los Angeles. For fastest service use cashier's check or money order. Add 3% for shipping and handling ($3.00 Min. max. East of Mississippi add 3%). Add 3% for Credit card orders. Prices and availability subject to change without notice. We reserve the right to limit quantities.

AUTHORIZED DEALER
TEXAS INSTRUMENTS

VISA MasterCard

24 Hour Order Line

(818) 366-6631

NOTE: Payment in full must accompany all orders. Credit card, company check or Money order for immediate shipment. Personal Checks require up to 4 weeks to clear. California residents add 6% for sales tax.

Trials of a c99 beginner

Two-dimensional floating-point arrays with subscripts starting at 0

By CHARLES E. KIRKWOOD JR.

The example in this article will use the function **twodim(n,i,j)** with two-dimensional floating-point arrays. The function **twodim(n,i,j)** in the March 89 issue was written for arrays with subscripts that start with one. In this article the array subscripts will start with zero, so a slight modification is made and the function renamed **td(n,i,j)**, where **n** is the number of columns of the two-dimensional array, **i** is the first subscript, and **j** is the second subscript.

```
/*function for arrays starting with 0*/
td(n,i,j)
int n,i,j;
{
  int k;
  k=n*i+j;
  return(k);
}
```

This month's example is the solution of simultaneous linear equations. Two functions are written, **simeq()** and **matinv()**, along with a main program for input and output. The matrix inversion method is used for the solution. Details for inverting a matrix can be found in numerical analysis books, so all of the details will not be discussed.

The usual method for a matrix inversion is to copy a unit matrix (ones on the diagonal starting at the upper left element with the other elements zero) after the given matrix, then swap the two matrices by arithmetic manipulations. A much shorter method is given by H. R. Meck in his book *Numerical Analysis with the TI/994A, Commodore 64, Apple II+/IIe and TRS-80 model I/III.* Instead of copying all the columns of the unit matrix, the first column follows the matrix to be inverted, the necessary calculations are made, the resulting matrix is shifted to the left, the second column of the unit matrix is copied, and the process is repeated. This is a considerable saving of memory.

The sample program is designed to solve up to ten simultaneous equations. More could be tried by increasing the dimensions of the arrays. The function **simeq()** prepares the input matrix **a** (each row of coefficients and the right-hand side) by copying the right-hand side into **c**. The matrix without the right hand side is then inverted by **invmat()**, and the results are calculated and stored into **x**.

```
/*Simultaneous Equations*/
#include DSK1.FLOATI
#extern printf(),atoi();
main()
{
  float a[110][8],x[10][8];
  int i,j,k,m,n;
  char s[20];
  puts("Input number of equations ");
  n=atoi(gets(s));
  m=(n)*(n+1);
  puts("\nInput rows, including right-hand side\n");
  j=0;
  k=0;
  for(i=0;i<m;++i)
  {
    printf("Input a(%d,%d) ",j,k);
    fpget(s,&a[i][0]);
    ++k;
    if(k==n+1)
    {
      ++j;
      k=0;
    }
  }
  simeq(n,a,x);
  for(i=0;i<n;++i)
    fpput(&x[i][0],s);
}

/*For solution of simultaneous equations*/
simeq(n,a,x)
float a[][8],x[][8];
int n;
{
  int i,j,z,m;
  float y[8],c[10][8];
  z=0;
  m=n+1;
  for(i=0;i<n;++i)
    fcpy(&a[td(m,i,n)][0],&c[i][0]);
  invmat(n,a);
  for(i=0;i<n;++i)
  {
    itof(z,&x[i][0]);
    for(j=0;j<n;++j)
    {
      fexp(&a[td(m,i,j)][0],"*",&c[j][0],y);
      fexp(&x[i][0],"+",y,&x[i][0]);
    }
  }
  return(x);
}

/*Matrix inversion*/
invmat(n,a)
float a[][8];
```

(See Page 25)

c99—

(Continued from Page 24)

```
int n;
{
 float y[8];
 int i,j,k,z,o,p,q,m;
 o=1;
 z=0;
 m=n+1;
 for(k=0;k<n;++k)
 {
   for(i=0;i<n;++i)
   {
     p=td(m,i,n);
     if(i==k)
       itof(o,&a[p][0]);
     else
       itof(z,&a[p][0]);
   }
   for(j=1;j<=n;++j)
   {
     p=td(m,k,j);
     q=td(m,k,z);
     fexp(&a[p][0],"/",&a[q][0],&a[p][0]);
     for(i=0;i<n;++i)
     {
       if(i!=k)
       {
         fexp(&a[p][0],"*",&a[td(m,i,z)][0],y);
         q=td(m,i,j);
         fexp(&a[q][0],"-",y,&a[q][0]);
       }
     }
   }
   for(i=0;i<n;++i)
   {
     for(j=0;j<n;++j)
       fcpy(&a[td(m,i,j+1)][0],&a[td(m,i,j)][0]);
   }
 }
 return(a);
}
```

After compiling and assembling, link with CSUP, FLOAT, and PRINTF.

Please pardon my goofs in the April c99 article. Change the first sentence in the third paragraph to "Adding two complex numbers, a+bi and c+di, gives (a+c)+(b+d)i ..." and the seventh statement of main() should be fgetcx(s,y).

The function **root()** in the August 1988 c99 article can be speeded up by initializing **ro** to **1.0** rather than **an;** that is, change **fcpy(an,ro);** to **fcpy(wone,ro);**.

USER GROUP UPDATE

The following are additions and updates to our user group listings, begun in May 1987.

California

Southern California Computer Group, P.O. Box 152535, San Diego, CA 92115 (new address).

TI-SIG, San Diego Computer Society, disbanded as of June 1. Most members can be contacted through the Southern California Computer Group.

Kansas

Mid/America 99 Users Group, c/o Shirley Slicer, 1101 Purdom St., Olathe, KS 66061. Phone: (913) 764-6451 (new address and phone). Meets 7-9 p.m. second Tuesday of each month. Disk, cassette, magazine and newsletter library. Annual dues $23 family, $20 individual.

Outside U.S.

Canada

B.C. 99er Users Group, 218-10th Ave., New Westminster, British Columbia, Canada V3L 2B2. PUBBS Data Line (604) 526-3389, WEBBS (604) 689-3227. Formerly New 99er Users Group. Approximately 45 members. Annual dues $30. Library (550 disks). Monthly newsletter, BBS access, hardware and software support. Meets at 7 p.m. Thursdays at Cameron Recreational Center, 9523 Cameron St., Burnaby, B.C. (First Thursday games, second Thursday general, third Thursday tutorials, fourth Thursday library night.)

THE MAKING OF A PORTABLE TI

32K, printer port and RAMdisk and it runs on batteries

By JAN JOEL JANOWSKI

I have succeeded in constructing a portable TI computer, and it is my desire to share my knowledge and experiences with all interested parties in the TI community. It has been a rewarding project, but not without its ups and downs. There were times that I felt like I was on an emotional roller coaster, as some problems were being solved while new ones were emerging on an almost nightly basis!

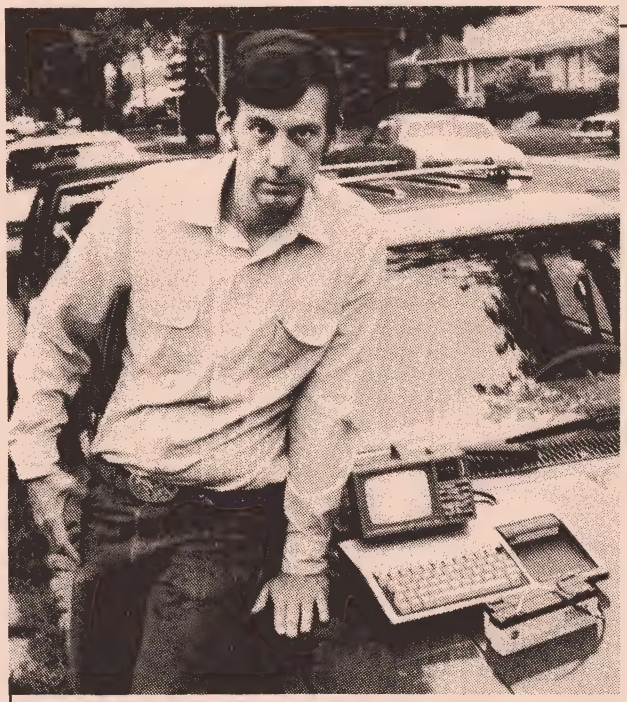
This project started out in June 1988 as an idea, or maybe *dream* would be a better word. This idea subsequently provided both the inspiration and impetus for my decision to attempt to create a portable TI-99/4A Home Computer. As I am now getting a little ahead of myself, I will now back up and start at the beginning. But before I begin, I must include the following disclaimer:

IMPORTANT WARNING

Failure to properly execute the instructions contained in this article will casue irreversible damage to a TI computer. Correctly following the instructions will lead you to make numerous permanent modifications to the internal components of your TI, and the reader alone will be responsible for the results. Neither the author or MICROpendium can accept responsibility for either the success or failure of any project, based upon this series of articles. All risks are to be assumed by the reader. It is recommended that readers who pursue this project obtain a black and silver TI99/4A console for use in this project and that they not attempt it until the entire 3-part series has been published.

In the beginning of this project, I first developed an eight-bank "super cart." I then determined that 32K of memory could be added to the 4A console. My next task was to find out if a battery could deliver sufficient power to my portable. Also, in the beginning, I knew that there was space to the left of the cartridge slot for the 32K of memory, as there had been various articles indicating this.

There are few reasons to have a portable computer unless you can do something with it while you are out "in the field." Just being able to create data is not enough; you want to be able to print it out, too. I then thought back to my purchase of my first Peripheral Expansion Box, for my TI. I had money for either a disk storage system or a printer and RS-232 card. After a considerable bit of arguing with myself, I chose the printer. To me, having the ability to output something to paper expands the power of the computer farther than having disk storage does. TI's subsequent reduction of the price of its disk storage system made a disk storage system more accessible to me, but I still feel that a printer expands the productivity of the TI far more than a disk system. (Also, the fact that my wife had just started a word processing service did not influence my arrival at this conclusion.)



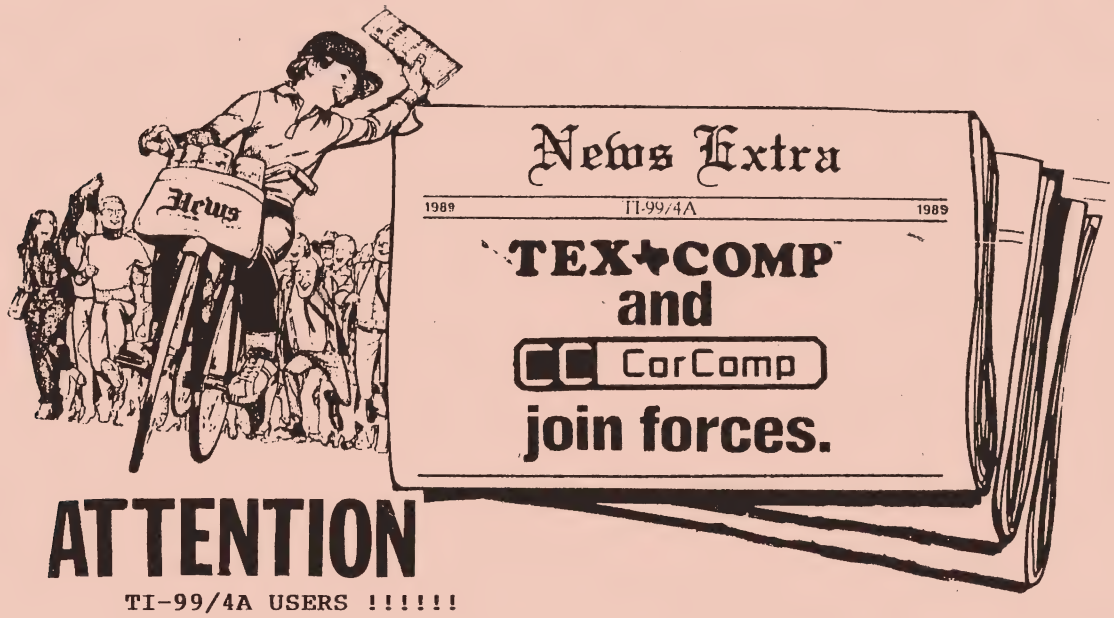
The author poses with his portable TI. Batteries are in the foreground. The unit includes 32K, RAMdisk and a printer port.

So, I was then convinced that I had to have (at least) a parallel port in the portable. In fact, my first design for the portable was based upon the idea of having a full 48K of memory, an 8- or 16-bank super cart, and a parallel port for outputting text to a printer. I still have that original "portable" computer attached to my HAM radio system, and it works quite nicely. Still, my first portable was obsolete as soon as it was finished, as it wasn't as flexible as a fully expanded TI; if I wrote something worth saving, there was no way to save it. The first portable worked as designed, but it was too restrictive relative to input and output capabilities, and I felt that an improvement was necessary.

The limitations that existed in my first design caused me to rethink the project and put more efforts into my experimentation. I then found that my time was being simultaneously split in different ways, and I was trying to improve various aspects of the portable's capabilities at the same time; I needed to be able to store information and recall it at will, also I did not have either the space or the current to power a disk drive, but I wanted *all* of the benefits that a disk storage system could give. I felt that I had set my standards too low on my first portable, and I was determined to not make that mistake again. My HAM radio computer subsequently became my "test bed" for the power supply that I had purchased at a Radio Shack store.

This power supply was clearly one that was designed for the TI99/A, but it *looked* better than any supply that I had ever seen

(See Page 28)



ATTENTION

TI-99/4A USERS !!!!!!

SINCE 1983 CorComp has been the leading name in 99/4A for reliability, dependability and value. CorComp is the only producer of TI-99/4A hardware to be sold by every major 99/4A supplier. In order to continue to be able to offer full support to the 99/4A Community, CorComp has instituted a "stocking dealer" program. Under this program, only those 99/4A dealers and distributors who are willing to demonstrate their support of the 99/4A by agreeing to stock sufficient levels of CorComp products will be be part of the CorComp Stocking Dealer Program and will continue to offer CorComp products. Tex-Comp has carried CorComp products since their introduction in 1983 and is proud to announce that it will continue to do so as an "Approved Stocking CorComp Dealer". This commitment of support to CorComp by Tex-Comp is also a commitment to the future of the TI-99/4A and of course to you the TI-99/4A user. The TI-99/4A is the one to start with and the one to stay with....

A COMPLETE EXPANSION SYSTEM FOR AS LOW AS $379.95

CorComp RS232 Card for TI P-Box..... 79.95
CorComp 32K Card for TI P-Box..... 99.95
CorComp 256K Card for TI P-Box..... WRITE
CorComp 512K Card for TI P-Box..... WRITE
CorComp 256K & 512K stand alone units for 9900 sys.. WRITE
CorComp DS/DD Disk Controller Card for TI P-Box.....149.95
CorComp 9900 expansion System.....299.95
(with 32K, RS232 and DS/DD Controller)
CorComp 9900 system with SS/SD disk drive.....379.95
with DS/DD drive (full or½ ht)..399.95
with pair of DS/DD ½ht drives...499.95
CorComp TRIPLE TECH Card (clock/64k buffer) 119.95
CorComp "grom buster" 1983 module adapter.....24.95
CorComp PDI diagnostic module.....24.95
CorComp Load Interrupt Switch & screen dump program..19.95
CorComp "IBM Connection" transfer ASCII Data.....39.95
CorComp "WRITER EASE +" word processor & spell check.39.95
Full size ss/sd drive..59.95 add $20 for ds/dd
½ height ds/dd drive (two will fit in P-Box.....89.95
Cable kit for two ½hts in P-Box $29.95 or $9.95 with 2 drives
Disk drive enclosure & power supply for 1 or 2 drives39.95

VISA MasterCard VISA and MASTERCARD HOLDERS CALL DIRECT (818) 366-6631 add 3% for credit card orders

SEND ORDER AND MAKE CHECKS PAYABLE TO TEX-COMP™ P O BOX 33084 — GRANADA HILLS CA 91344

Texas Instruments AUTHORIZED DEALER

CorComp AUTHORIZED SALES

TERMS: All prices F.O.B. Los Angeles. For fastest service use cashiers check or money order. Add 3% shipping and handling ($3.00 minimum). East of Mississippi 4½% (Free shipping on all software orders over $100.00). Prices and availability subject to change without notice. We reserve the right to limit quantities

NOTE: Payment in full must accompany all orders. Credit Card, Company Check or Money Order for immediate shipment. Personal checks require up to 4 weeks to clear. California orders add 6½% sales tax.

"The Leader of the Pack"

PORTABLE TI—

(Continued from Page 26)

seen in a black TI console before. Yes, it just *looked* better. The power connection plug had only two wires coming from it, and the words "IRIICHI TSUSHIN KOGYO CO. LTD." and the part number, 1053214-2, were stamped on it. It looked like a "switching" power supply. Please allow me a short digression, while I comment on the nature of "switch mode" or "switching" power supply.

SWITCHING POWER SUPPLIES

Switching power supplies work more or less like this: A power supply creates a voltage, which drives an oscillator, which drives a transformer. The primary and secondary, in a transformer are isolated from one another, and a diode on the secondary side converts the A.C. pulses to D.C. pulses, which are smoothed by capacitors and inductors.

The most significant thing that I noticed was that this power supply had only two wires for input from the power source. This meant that a common supply was being used to create the plus and minus power output. This was a very significant factor, as the use of only two wires meant that it could be powered by a battery. Now, all this was well and good, but the essential question was: Will it power a TI console? I then tested it on the TI console which I was using exclusively for my HAM radio, and all went well. In fact, it was much cooler under extended periods of use than the original power supply.

At this point, I started doing two things: First, I sent a letter to TI, asking about the power supply that I had stumbled on, and, second, I started testing one of the supplies to see literally how much current it would supply before failing. (I started immediately with the second activity only because I never thought I would receive an answer from TI.) I ran it for an hour at 3.2A, and, though it got very warm, it did not fail. Then, to my surprise, my letter was answered by TI, complete with a schematic diagram and a theory of its operation. Accompanying all this was a letter saying that I could not photostat it! Okay, now you know why I can't provide you with a copy, but I can tell you the details of what TI was kind enough to send me.

According to TI, this "QI version power supply" is rated at 1.6A on the +5V supply. (Tests had shown that a "stock" TI draws 1.0A, but a TI with 32K memory expansion (4 8K chips) and a Hamsoft Module, draws 1.04A, and that was within the limits of this power supply's rated output. I then tried running this supply on 12V D.C. It would work, but I was not happy with it. This is because the current drain was too great for the computer to operate for very long on the battery.

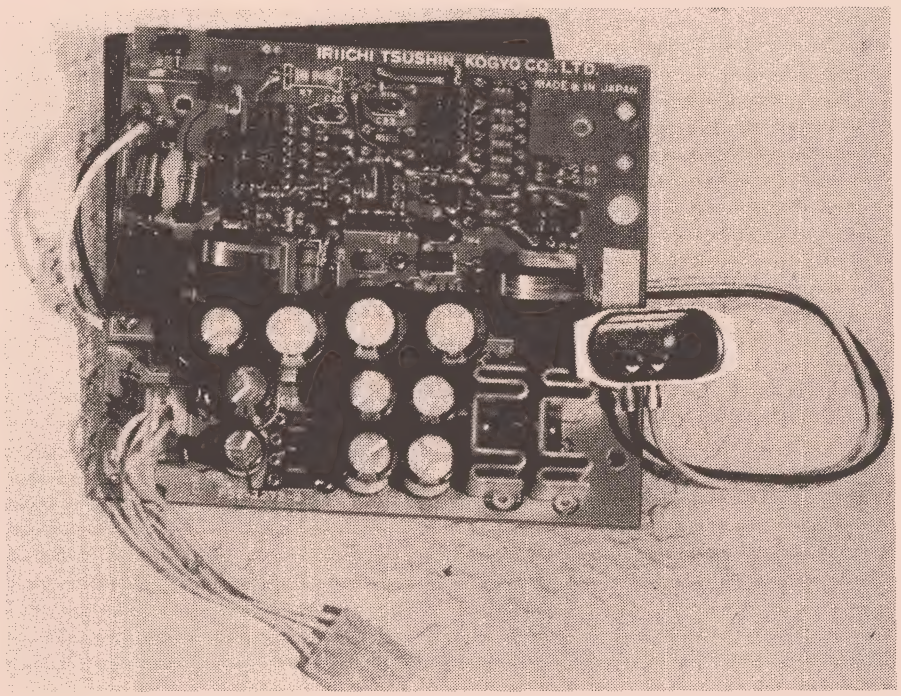


Fig. 1 "QI" revision power supply is used to make a TI battery operated.

I then went back to A.C. power and took some voltage measurements, to find a way around the current problem. The power before the switching oscillator (after the full wave bridge rectifier) was at 27V D.C., so I increased my supply voltage from 12V to 24V, and I found that the power supply worked just fine.

But the question remained: Would it power the TI or damage it? To my surprise, it worked fine. In the course of further testing, I determined that I could operate my TI, with 32K of memory and a Hamsoft Module and the 8-bank supercart, off two 1.9AH 12V batteries in series, for well over two hours without a problem. I could even charge them when the AC supply output was running the TI! Sweet success at last. (See Fig. 1.)

At this point, I was certain that I did not want a portable with only a multi-bank "super cart" and no storage capacity. I therefore committed myself to building a second portable with a RAMdisk inside the console. But a commitment based upon an intent and a desire to do something is not enough. Rather, you must first verify the feasibility of your idea before you even start.

Relative to the execution of my idea, I thought that I had done my homework, which was to ask someone to try to load files from a Horizon RAMdisk (HRD) without a disk controller card in his TI PEB. This he was able to do without difficulty. Little did I know that what I initially thought of as sufficient homework, my R&D, was a gross oversimplification of the real problems that this project would entail, and that oversimplification would come back to haunt me later.

My plan revolved around an "expansion board" that would reside inside the console. In all honestly, I was unsure of where I should start. I knew that if I did not plan my steps correctly, I could well find myself halfway done, and in a corner with no way to proceed. My biggest worry was that I

(See Page 29)

PORTABLE TI—

(Continued from Page 28)

would make a wiring error while I was in the testing stage, and the computer would totally lock up, and I would not be able to do any troubleshooting or testing because the computer was locked up — a no-win situation.

From that scary idea came my plan of attack: The expansion board would have to be able to be disconnected from the computer, so that it could be tested separately. I therefore decided on a "header plug" to do the interconnecting of the ribbon cable from the GROM socket and the extra wires from the computer itself. To make my expansion board easily separable for testing and to make its wiring independent of the computer made good sense to me. Also, the computer would be less apt to be zapped by static electricity when disconnected from the expansion board. The decision to use the header also meant that a wiring interconnect diagram had to be made so that a schematic of the interconnect from the computer to the expansion board could be made. I at first thought that 50 pins would be sufficient, but I quickly determined that 50 wasn't enough, so used 60 pins, as this is the next higher number which would be easily available for use with a flat wire cable. The layout of the 60-pin header is described below.

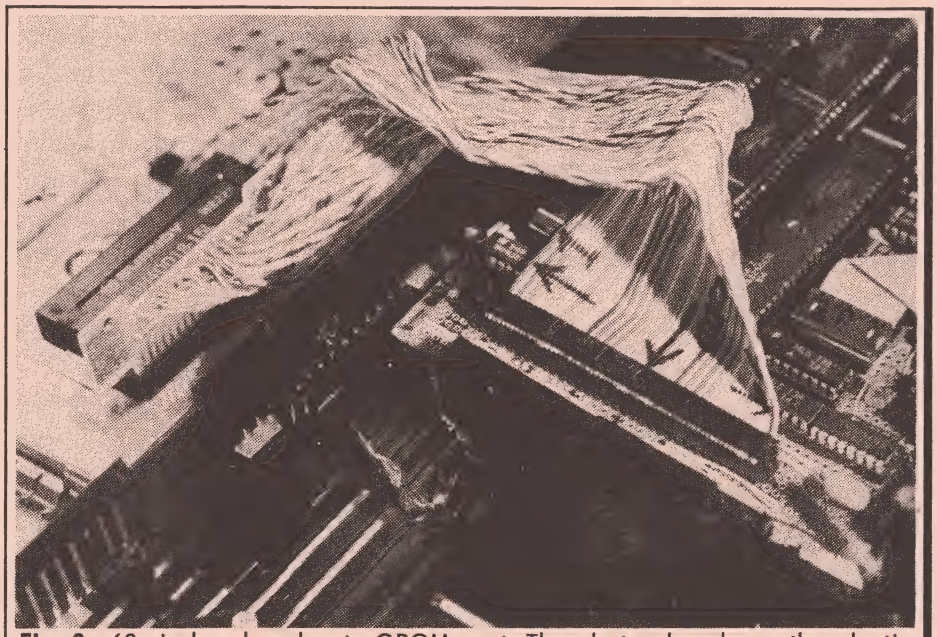


Fig. 2 60-pin header plug to GROM port. The photo also shows the plastic that was cut away behind the GROM port in the top of the cover.

LAYOUT OF 60-PIN HEADER

Pins 1 through 36 are identical to the grom connector.
37. U504 pin 14 (2000-3FFF)
38. U504 pin 10 (A000-BFFF)
39. U504 pin 9 (C000-DFFF)
40. U504 pin 7 (E000-FFFF)
41. U508 pin 9 (Not DBIN)
42. U510 pin 18 (A2)
43. U510 pin 16 (A1)
44. U510 pin 14 (A0)
45. U605 pin 3 (Not MEMEN)
46. U508 pin 6 (Not DLY RST)
47. pin 4 of I/O connector (Not EXT INT)
48. U510 pin 12 (Not phase 3 CLK)
49. U510 pin 3 (Not MBE) Pins 50 through 60 were not connected yet, for the purpose of expansion. (It was later determined that you can disregard pins 46-49 totally.)

In order to connect these wires to the computer, an access hole had to be cut in the top metal cover of the computer, adjacent to the slot where the GROM connector plugs into the chassis. These wires are connected to the expansion board via the header plug, which contained a dual row of 30 pins with 0.1-inch spacing, for mounting on one end of a vector board that has solder pads on it. In this way, I could easily connect or disconnect the expansion board for testing.

Pins 37 through 41 are used for the 32K expansion (more on that later). Pins 42 to 45 are for the RAMdisk inside the console. Pins 46 through 49 are needed for the parallel interface (via Kantronics Hamsoft module). Not knowing what I eventually would need, I gave myself 11 spare connections, and they are taped inside the computer.

I finally reached the starting point of the construction. The experimental console was torn apart, and the power supply exchanged and tested. The ribbon cable connections were made to the computer board and GROM connector. The other end of this ribbon cable was fitted with a 60-pin header, and the entire cable was checked for continuity. Finally, the computer was tested to verify 100 percent proper operation and then placed on the shelf to wait for the completion of the expansion board.

The supports directly behind the GROM connector of the top cover were cut away so that the wires from the rear of the GROM connector to the 60-pin header plug would clear the plastic mounting assembly. The inside top cover of the console was modified so that the metal tabs from the chrome trim would not short out the expansion board. Carefully, a piece of vector board with solder tabs on both sides was cut to fit inside the upper cover of the console snugly between the left side of the console to the GROM connector mounting assembly. The expansion board ends up being approximately 5 inches x 9.75 inches, with some cutouts and notches for the supports for the mounting posts. A 5-inch by 9.75-inch board is all that I could fit in the small space inside the console, and on that board I wanted to put a 512K RAMdisk, a 32K memory expansion, a parallel port, and have sufficient room remaining for expansion. (See Fig. 2.)

Part 2 of this 3-part series on constructing a portable TI will be published next month.—Ed

A CHARACTER GENERATOR FOR GENEVE & TI99/4A

Assembly program creates and saves CHARA1 files

By WAYNE STITH

At one time or another, most of us have wished we could alter the CHARA1 files which accompany the programs we use. Many of us have tortured ourselves with a sector editor and changed the character patterns directly on disk, and some of us have even typed in an entire character set by hand to fit in the GRAM Kracker. I have wondered why there was not a simple program which would do the dirty work for us. Even my own KWIKFONT character definition program (see review, January, 1988) would not do this without some judicious fiddling.

Well, put your sector editor away! CHARA1FIX will turn a tedious chore into a quick delight. The concept of the program is so blazingly simple that I am puzzled why there are not dozens of such programs already floating around. CHARA1FIX loads in a CHARA1 file, lets you alter each character pattern by either typing in the infamous 16-byte definition string or manipulating *pixels* in a grid, and then saves the file back to disk. It is a quick and dirty approach which can easily be expanded to include any features you care to write, as the accompanying source code shows.

CHARA1 files are memory image (PROGRAM) files like EDIT1, ASSM1, etc. A normal EA5-type loader finds the load address in the 6-byte header and slavishly places the file in the pattern descriptor table which is usually located at >0800 in VDP. But within an assembly program one can ignore the load address and place the file in any convenient location. By placing CHARA1 elsewhere we avoid disturbing the current patterns.

Once the user has entered a drive number, CHARA1 is loaded into VDP and copied to CPU RAM so that pattern manipulations can be carried out quickly. CHARA1 is again accessed via a DSR subroutine which reads in the file's parameters. The information we need is the number of sectors the file occupies, not counting the file header. Normally this will be 4 or 8 sectors, depending on whether you are using the short or long CHARA1 file.

CHARA1FIX will let you redefine only the characters originally contained in the file, i.e. the 4-sector version will cover ASCII 0-127 while the longer version covers the entire character set. The highest usable character is calculated on the basis of the number of sectors used.

There are two glitches in CHARA1 files which deserve mention. The 6-byte header of the long file indicates that >0800 bytes (2K) should be loaded into VDP. Unfortunately, this information is also present in the 6-byte headers of some short files as well. CHARA1FIX disregards this header and creates its own based on the number of sectors actually used in the original file.

But the real glitch is the fact that the last character defined by CHARA1 is not fully defined. The second word of the 6-byte header contains the number of bytes to load, but unfortunately this includes the header. This means that the last character to be defined is short-changed by 6 bytes. The entire character set should occupy 8 sectors by itself and an additional sector should cover the extra data. The whole file would catalogue as 10 sectors. But TI apparently considered ASCII 255 to be useless, since many programs overwrite the last part of the character set anyway. CHARA1FIX follows the TI conventions here, as you will notice when you doublecheck a CHARA1 file you have created.

The main portion of the program should be thought of as a skeleton which holds the whole thing together. The dirty work is done in the routines called by BLWP's. The skeleton loads CHARA1, saves it back to disk, and checks the keyboard.

CHARA1FIX contains no fancy mirroring, reversals or insertions, but it is easily expandable by the addition of new routines to the skeleton. Write your code so that it can be accessed with BLWP. Since this program is similar to KWIKFONT, several of the routines in that program could be ported over with small modifications.

Because of the length of the CHARA1FIX program, a portion of it it will be published this month with the remainder to follow next month. The program should be entered using the Editor/Assembler editor or MY-Word in the Program Edit mode.—Ed

CHARA1FIX

```
      DEF  START,SFIRST,SLOAD,SLAST

SLOAD
SFIRST B   @START

REGS   EQU  >8318           Fast RAM for main workspace
INTREG EQU  >2000+32        Normal RAM for interrupt workspace
SET    DATA >2000           Constant
MSG1   TEXT 'CHARA1FIX'
MSG2   TEXT 'A utility for'
MSG3   TEXT 'altering CHARA1 files'
MSG4   TEXT '<Press any key>'
MSGH   TEXT 'FCTN-7 for help'
MSG5   TEXT 'Enter drive: DSK1.CHARA1'
MSG6   TEXT 'I/O error!'
MSG7   TEXT '  Ready to write to DSK1.CHARA1.'
MSG8   TEXT '  Press <ENTER> to confirm,'
MSG9   TEXT '  or any other key to abort!'
MSGA   TEXT 'CHR$('
MSGB   TEXT '(C) 1989 by Wayne Stith'
MSGC   TEXT 'Revision 3'

HMSG1  TEXT '  This program will allow you   '
```

(See Page 31)

CHARA1FIX—

(Continued from Page 30)

```
       TEXT '  to easily alter a CHARA1 file.'
       TEXT '  At the prompt, enter the      '
       TEXT '  number of the drive in which  '
       TEXT '  CHARA1 can be found. CHARA1   '
       TEXT '  will be loaded into memory.   '
       TEXT '  You will then see a screen    '
       TEXT '  which contains the following  '
       TEXT '  information: number of the    '
       TEXT '  character on which you are    '
       TEXT '  currently working, which is   '
       TEXT '  CHR$(Ø) at first; the charac- '
       TEXT '  ter pattern as it currently   '
       TEXT '  appears in normal size; the   '
       TEXT '  hex value for the pattern;    '
       TEXT '  and the character displayed   '
       TEXT '  in a large box.               '
       TEXT '                                '
       TEXT '       <Press any key>          '
       TEXT '                                '

HMSG2  TEXT '  You have several choices:     '
       TEXT '                                '
       TEXT '  (1) Alter the pattern in the  '
       TEXT '      box                       '
       TEXT '  (2) Alter the hex display     '
       TEXT '  (3) Select another character  '
       TEXT '  (4) Restore original charac-  '
       TEXT '      ter pattern               '
       TEXT '  (5) Write data to CHARA1      '
       TEXT '  (6) Load a new CHARA1 file.   '
       TEXT '                                '
       TEXT '  In the box use FCTN or CNTL   '
       TEXT '  arrows to move. <ENTER>       '
       TEXT '  will toggle the "pixel" on or '
       TEXT '  off. On the hex line type in  '
       TEXT '  the desired code. Use arrows  '
       TEXT '  to move. FCTN-9 toggles be-   '
       TEXT '  tween hex line and box.       '
       TEXT '                                '
       TEXT '       <Press any key>          '

HMSG3  TEXT '  Select another character by   '
       TEXT '  pressing FCTN-4 to go back or '
       TEXT '  FCTN-6 to go ahead.           '
       TEXT '                                '
       TEXT '  Restore original pattern for  '
       TEXT '  the current character by      '
       TEXT '  pressing CNTL-R.              '
       TEXT '                                '
       TEXT '  Write the data back to the    '
       TEXT '  original file by pressing     '
       TEXT '  CNTL-W. You will be asked to  '
       TEXT '  confirm your action.          '
       TEXT '                                '
       TEXT '  FCTN-7 will activate these    '
       TEXT '  screens again.                '
       TEXT '                                '
       TEXT '  FCTN-5 will allow you to load '
       TEXT '  another CHARA1 file.          '
       TEXT '                                '
       TEXT '       <Press any key>          '

HMSG4  TEXT '  Whenever you make a change to '
       TEXT '  either the hex pattern or the '
       TEXT '  box, the changes will be re-  '
       TEXT '  flected in the hex display,   '
       TEXT '  in the box, and in the small  '
       TEXT '  character.                    '
       TEXT '                                '
       TEXT '  Note that characters are shown'
       TEXT '  in graphics (8x8) mode. Text  '
       TEXT '  mode will ignore the right two'
       TEXT '  columns. The program is aware'
       TEXT '  of the length of the CHARA1   '
       TEXT '  file and will not allow you   '
       TEXT '  to alter characters which     '
       TEXT '  would fall outside its        '
       TEXT '  boundaries. For GENEVE users, '
       TEXT '  CNTL-= will quit to the       '
       TEXT '  title screen.                 '
       TEXT '                                '
       TEXT '       <Press any key>          '

NAME   TEXT 'CHARA1    '
DRIVE  BYTE Ø            Stores drive number

CODES  BYTE >Ø1          FCTN7
       BYTE >ØD          <ENTER>
       BYTE >Ø8          Left
       BYTE 147          Left
       BYTE >Ø9          Right
       BYTE 132          Right
       BYTE 11           Up
       BYTE 133          Up
       BYTE 1Ø           Down
       BYTE 152          Down
       BYTE 2            FCTN4
       BYTE 12           FCTN6
       BYTE 151          CNTLW
       BYTE 15           FCTN9
       BYTE 14           FCTN5
       BYTE 146          CNTLR
       BYTE Ø            Marks end of list

SPLIT  BYTE 157          Test value for leaving program

HEX    TEXT 'Ø123456789ABCDEF'
       EVEN

JMPTB1 DATA FCTN7,ENTER,LEFT,LEFT       Branch-address table #1
       DATA RIGHT,RIGHT,UP,UP
       DATA DOWN,DOWN,FCTN4,FCTN6
       DATA CNTLW,FCTN9,START,CNTLR,Ø

JMPTB2 DATA F7,SCAN2,LL,LL              Branch-address table #2
       DATA RX,RX,SCAN2,SCAN2
       DATA SCAN2,SCAN2,F4,F6
       DATA CNTLW,F9,START,CNTLR,Ø

CPAB   DATA >Ø5ØØ,>2ØØØ-6,Ø,>1ØØØ,>ØØØB      Input PAB
       TEXT 'DSK1.CHARA1'
       EVEN

CPAB1  DATA >Ø114     PAB data for direct-file-input routine
*                     (Name is >14, length of name is 1)

CPAB2  DATA >Ø6ØØ,>2ØØØ-6,Ø,Ø,>ØØØB         Output PAB
       TEXT 'DSK1.CHARA1'
       EVEN
```

(See Page 32)

CHARA1FIX—

(Continued from Page 31)

```
CURCHR DATA Ø Holds number of current character
TOTSEC DATA Ø Holds number of sectors in file CHARA1
HIGHCH DATA Ø Holds number of highest character we can fiddle with
FOUR   DATA 4 Constant
FIVE   DATA 5     "
SEVEN  DATA 7     "
EIGHT  DATA 8     "
HUND   DATA 1ØØ   "
TEN    DATA 1Ø    "

X32    DATA 32    "
X256   DATA 256   "
MODE   DATA Ø     Flag for grid/hex modes

CURSOR DATA Ø,Ø,Ø,>ØØFF    Cursor definition

ENVIR  DATA >ØØØØ,>Ø1EØ,>Ø2ØØ,>Ø3ØE   Data for VDP registers
       DATA >Ø4Ø1,>Ø5Ø6,>Ø6ØØ,>Ø713

LINE#  DATA Ø             Line number in the grid

* Visual representation of all 16 possible combinations of '1' and 'Ø'
* in four bits.  ASCII Ø will represent an off pixel, and ASCII 8 will
* represent a pixel which is turned on.

BOXDAT BYTE Ø,Ø,Ø,Ø      ØØØØ
       BYTE Ø,Ø,Ø,8      ØØØ1
       BYTE Ø,Ø,8,Ø      ØØ1Ø
       BYTE Ø,Ø,8,8      ØØ11

       BYTE Ø,8,Ø,Ø      Ø1ØØ
       BYTE Ø,8,Ø,8      Ø1Ø1
       BYTE Ø,8,8,Ø      Ø11Ø
       BYTE Ø,8,8,8      Ø111

       BYTE 8,Ø,Ø,Ø      1ØØØ
       BYTE 8,Ø,Ø,8      1ØØ1
       BYTE 8,Ø,8,Ø      1Ø1Ø
       BYTE 8,Ø,8,8      1Ø11

       BYTE 8,8,Ø,Ø      11ØØ
       BYTE 8,8,Ø,8      11Ø1
       BYTE 8,8,8,Ø      111Ø
       BYTE 8,8,8,8      1111
       EVEN

START  LWPI REGS          Load WS
       CLR  @>83C4        Clear ISR hook in case program returns here

       LI   R13,>8CØ2     VDP write-address 'port'
       LI   R14,>8CØØ     VDP write-data 'port'
       LI   R15,>88ØØ     VDP read-data 'port'

       CLR  @MODE         Set mode to Ø (Ø = grid mode)
       BL   @CS           Clear screen

* Set up environment by loading VDP registers

       LI   R1,ENVIR
SETUP  MOV  *R1+,RØ
       BLWP @VWTR
       CI   R1,ENVIR+16
       JNE  SETUP
       MOVB @ENVIR+3,@>83D4   Save new contents of VDP R1

* Change color of color sets

       LI   RØ,>38Ø
       LI   R1,>13ØØ      >13 = black on light green
SETUP1 BLWP @VSBW
       INC  RØ
       CI   RØ,>38Ø+32
       JNE  SETUP1

* Redefine the cursor pattern

       LI   RØ,>8FØ       (3Ø*8+>Ø8ØØ)
       LI   R1,CURSOR
       LI   R2,8
       BLWP @VMBW

* Set up ASCII Ø and 8 as white and black "pixels" respectively

       LI   RØ,>8ØØ
       LI   R1,>FFFF      All pixels on
       LI   R2,16
SETY   BLWP @VSBW
       INC  RØ
       DEC  R2
       JNE  SETY

* Now change the color sets for sets Ø and 1

       LI   RØ,>38Ø
       LI   R1,>FF11      White
       BLWP @VSBW

       INC  RØ
       SWPB R1            Black
       BLWP @VSBW

* Messages on introductory screen

       LI   RØ,299
       LI   R1,MSG1
       LI   R2,9
       BLWP @VMBW

       LI   RØ,299+32+3Ø
       A    R2,R1
       LI   R2,13
       BLWP @VMBW

       LI   RØ,299+32+32+32+26
       A    R2,R1
       LI   R2,21
       BLWP @VMBW

       LI   RØ,745
       A    R2,R1
       LI   R2,15
       BLWP @VMBW

       LI   RØ,484
       LI   R1,MSGB
       LI   R2,23
       BLWP @VMBW
```

(See Page 33)

CHARA1FIX—

(Continued from Page 32)

```
       LI   R0,555
       LI   R1,MSGC
       LI   R2,10
       BLWP @VMBW

PASS   BL   @KK          Wait for a keypress

PASS1  BL   @CS          Clear screen

       LI   R0,162
       LI   R1,MSG5      'Enter drive...'
       LI   R2,24
       BLWP @VMBW

* Set up interrupt routine for blinking cursor

       LI   R0,178       Screen location
       LI   R1,>3100     '1'
       LI   R3,MYINT1
       MOV  R3,@>83C4          Load ISR hook
       MOV  @FIVE,@INTREG+20   Time delay value

PASS2  BL   @KK          Wait for a key
       CLR  R3
       MOVB @>8375,R3    Fetch key value
PASS2B CI   R3,>0D00     <ENTER> ?
       JEQ  PASS2A       Yes
       CI   R3,>3100     '1' ?
       JL   PASS2        No, too little, scan again
       CI   R3,>3900     '9' ?
       JH   PASS2        No, too high, scan again
       MOV  R3,R1        Key OK
PASS2A BLWP @VSBW        Put on screen
       CLR  @>83C4       Kill ISR hook
       MOVB R1,R4        Store value
       MOVB R1,@DRIVE    Save drive number (with mask)

       LI   R0,>3500     Send PAB to VDP
       LI   R1,CPAB
       LI   R2,21
       BLWP @VMBW

       MOVB R4,R1        Fetch drive number
       LI   R0,>3500+13  Point to location of drive in PAB
       BLWP @VSBW        Insert new drive number in PAB

       LI   R0,>3509     Point >8356 to the PAB length byte
       MOV  R0,@>8356
       BLWP @DSRLNK      Fetch CHARA1 file
       DATA 8

       JNE  PASS3        If no error, skip ahead

PASSX  BL   @CS          Clear screen
       LI   R0,299
       LI   R1,MSG6
       LI   R2,10
       BLWP @VMBW        Print error message

       LI   R0,745
       LI   R1,MSG4      'Press any key...'
       LI   R2,15
       BLWP @VMBW
       BL   @KK          Wait for keypress
       B    @START       Restart program

* File load successful, determine number of sectors by using
* direct-file-input subroutine for parameters

PASS3  CLR  @>834C       >834D must be clear for paramter purposes
       AI   R4,->3000    Eliminate the >30 mask from drive number
       MOVB R4,@>834C    Put drive number at >834C
       LI   R0,>3000     >834E must contain the VDP address where the
       MOV  R0,@>834E       filename can be found (minus 'DSKx.')
       LI   R1,NAME      Put the 10-byte name CHARA1 at >3000; it
       LI   R2,10           will have trailing spaces
       BLWP @VMBW
       CLR  @>8350       If zero, then parameters are returned at >8300-->

       LI   R0,>3500
       LI   R1,CPAB1
       LI   R2,2
       BLWP @VMBW        Send the PAB to VDP

       LI   R0,>3500
       MOV  R0,@>8356    Pointer to length byte
       BLWP @DSRLNK      Read in file parameters
       DATA 10           Use 10 for subroutine access
       JEQ  PASSX        If error, use routine above

* Subroutine >14 places the number of sectors the file occupies (not
* counting the header) at >8302.

       MOV  @>8302,@TOTSEC    Save number of sectors

* Determine the highest character we can play with: Multiply the
* number of sectors by 256 to obtain the number of bytes in the file;
* then subtract 6 bytes to account for the 6-byte header information.
* Then divide by 8 to determine the number of characters this
* represents, which will also be the number of the highest character.

       MOV  @TOTSEC,R0
       MPY  @X256,R0
       AI   R1,-6        Answer will be only in R1
       CLR  R0           Clear for division
       DIV  @EIGHT,R0
       MOV  R0,@HIGHCH   Save the number of the highest character usable

       CLR  @CURCHR      Set current character to ASCII 0

* CHARA1 was not loaded to its normal place in VDP, but rather to
* >2000-6. The actual data begin at >2000.

       LI   R0,>2000     Move data to a RAM buffer
       LI   R1,>E000
       LI   R2,>1000
       BLWP @VMBR

TRY    BL   @CS          Clear screen

* Show main screen messages

       LI   R0,11
       LI   R1,MSG1      'CHARA1FIX'
       LI   R2,9
       BLWP @VMBW

       LI   R0,745
```

(See Page 34)

CHARA1FIX—

(Continued from Page 33)

```
       LI   R1,MSGH      'FCTN-7...'
       LI   R2,15
       BLWP @VMBW

* Display 'CHR$( '

       LI   R0,163
       LI   R1,MSGA
       LI   R2,5
       BLWP @VMBW

* Display the number of the current character

MAIN1  LI   R6,>FE00     Use >FE00 as a little buffer
       CLR  R11          Flag; if zero, we have leading zeros in answer
       MOV  @CURCHR,R5   Fetch current character
       CLR  R4           Clear for division

* (After division, R4 will contain the integer result, while R5 will
* contain any remainder.)

       DIV  @HUND,R4     Divide by 100
       MOV  R4,R4        Zero ?
       JEQ  CONV1        Yes, skip ahead
       SWPB R4           Place integer in high byte
       AI   R4,>3000     Add mask of >30 to display it
       MOVB R4,*R6+      Store
       SETO R11          Set flag

CONV1  CLR  R4
       DIV  @TEN,R4      Divide by 10
       MOV  R4,R4        Zero ?
       JNE  CONVA        No
       MOV  R11,R11      If flag is still zero, we skip ahead
       JEQ  CONV2        because the hundreds value was zero as well

CONVA  SWPB R4           Place integer in high byte (tens' value)
       AI   R4,>3000     Mask
       MOVB R4,*R6+      Store

CONV2  SWPB R5           Fetch remainder, which is also the ones' value
       AI   R5,>3000     Mask
       MOVB R5,*R6+      Store

       LI   R1,') '
       MOVB R1,*R6+      Add right parenthesis

CONV3  MOVB @SET,*R6+    Clear out rest of this little buffer
       CI   R6,>FE00+3
       JLE  CONV3

       LI   R0,168       Add the character number plus ')' to screen
       LI   R1,>FE00
       LI   R2,4
       BLWP @VMBW

       BLWP @BOXWRT      Display the grid
       BLWP @LITTLE      Display the little character
       BLWP @HEXDIS      Display the hex string
       CLR  @LINE#       Point to line 0 of grid

* Are we in hex mode or grid mode ?

       MOV  @MODE,@MODE  If zero, then grid mode
       JEQ  MAINX
       B    @MODE2       Nope, hex display mode

* Main routines for working the grid

MAINX  LI   R0,114  Point to top left corner of grid
PRESC1 BLWP @VSBR   Read the character

       MOV  @FIVE,@INTREG+20  Set interrupt timer delay
       LI   R3,MYINT1
       MOV  R3,@>83C4    Load ISR hook

SCAN1  BL   @KK          Scan for a key
       BLWP @VSBW        Display current value in R1
       MOVB @>8375,R3    Fetch key

* Check for acceptable codes

       LI   R6,CODES     Point to code table
       LI   R7,JMPTB1    Point to branch-address table

CHECK  MOVB *R6+,R2      Fetch code byte to check
       JEQ  SCAN1        If the byte was zero, end of list; rescan

       CB   R3,R2        Code match ?
       JEQ  CHECK1       Yes
       INCT R7           Add 2 to address pointer because it's DATA
       JMP  CHECK        Check some more

CHECK1 MOV  *R7,R7       Fetch address into R7
       B    *R7          Branch to that address

* Move cursor left.  By multiplying the grid line number (0-7) by 32, we
* come up with a screen line number.  Adding 114 adjusts this value to
* point to the leftmost spot in the grid on that grid line.

LEFT   MOV  @LINE#,R5
       MPY  @X32,R5
       AI   R6,114

* R0 is the cursor location.  If it is at the far left, it must wrap
* around the grid and appear at the far right on the line above.

       C    R0,R6        At the far left ?
       JEQ  LEFT1        Yes

       DEC  R0           No, decrement cursor location and start over
       JMP  PRESC1

LEFT1  DEC  @LINE#       Subtract 1 from grid line pointer
       AI   R0,-25       Place cursor at far right on previous line
       CI   R0,114       Out of the grid ?
       JHE  PRESC1       No, start over

       LI   R0,345         Yes, point cursor to bottom right corner
       MOV  @SEVEN,@LINE#  Set grid line pointer to 7 (bottom line)
       JMP  PRESC1         Start over

* Move right. Again, multiply to determine the screen line, but then add
* 121 (114+7) to determine the rightmost location on the grid line.

RIGHT  MOV  @LINE#,R5
       MPY  @X32,R5
       AI   R6,114+7
       C    R0,R6        At the far right ?
       JEQ  RIGHT1       Yes

       INC  R0           No, increment cursor location and start over
       JMP  PRESC1
```

The remainder of this program will be published next month.

Calendar maker

Print a year of months on a single page

The following program is a companion to the Calendar Maker program that was published last month. While last month's program printed a calendar for any month, this program prints a calendar for an entire year on a single 8½ x 11 page.

The program is by Dale A. Kloes of Gibsonia, Pennsylvania. It runs in Extended BASIC. Included with the program is a menu that allows the user to load two other calendar programs — the one that appeared last month and a third that displays the calendars on the screen rather than outputting them to a printer. Because of its length, the third program may not be published.

CALENDAR 3

```
10 REM CALENDAR3 - PRINT YEA
R ON PRINTER !115
20 REM (C) 1983-88 BY DALE A
. KLOES     PUBLIC DOMAIN !02
1
30 CALL CLEAR !209
40 DEF INVERT(A)=A-INT(A)!07
7
50 DIM HEAD$(5)!043
60 DIM DOTW$(11,5,6)!232
70 GOSUB 1060 !120
80 YRMSG1$="ENTER THE YEAR"
:: PFLG$="Y" !248
90 YRMSG2$="YEAR MUST BE 158
3 THRU 9999" !033
100 YRMSG3$="NO. OF COPIES"
!222
102 DISPLAY AT(1,1):"
  PRINT YEAR" !051
104 DISPLAY AT(2,1):"(C) 198
3-88 BY DALE A. KLOES" !131
106 DISPLAY AT(3,1):"
PUBLIC DOMAIN" !175
107 IF PFLG$="N" THEN 110 !1
40
108 PFLG$="N" :: DISPLAY AT(
4,1):"ENTER PRINTER:" !210
109 ACCEPT AT(5,1)BEEP:PRNTR
$ !009
110 DISPLAY AT(6,1):YRMSG1$
:: ACCEPT AT(6,23)VALIDATE("
1234567890")BEEP SIZE(4):YR$
 !187
120 IF VAL(YR$)>=1583 THEN D
ISPLAY AT(24,1):" " :: GOTO
140 !059
130 DISPLAY AT(24,1):YRMSG2$
 :: GOTO 110 !184
140 DISPLAY AT(8,1):YRMSG3$
:: ACCEPT AT(8,23)VALIDATE("
1234567890")BEEP SIZE(2):COP
YNO !200
150 CN=VAL(SEG$(YR$,1,2))!00
5
160 YR=VAL(SEG$(YR$,3,2))::
OLDYR=YR !150
170 HEAD$(5)=SP3$&YR$&SP27$&
"  CALENDAR  "&SP27$&YR$ !12
4
180 FOR MONTH=1 TO 12 !168
190 MN$=" " !101
200 ON MONTH GOTO 210,260,36
0,400,430,460,490,520,550,58
0,610,640 !090
210 MN=11 !133
220 YR=YR-1 !208
230 MN$="J" !143
240 MX=31 !145
250 GOTO 660 !229
260 MN=12 !134
270 YR=OLDYR !243
280 IF INVERT(YR/4)<>0 THEN
320 !103
290 MX=29 !152
300 IF YR<>0 THEN 330 !107
310 IF INVERT(CN/4)=0 THEN 3
30 !150
320 MX=28 !151
330 YR=YR-1 !208
340 MN$="F" !139
350 GOTO 660 !229
360 MN=1 !083
370 YR=OLDYR !243
380 MX=31 !145
390 GOTO 660 !229
400 MN=2 !084
410 MX=30 !144
420 GOTO 660 !229
430 MN=3 !085
440 MX=31 !145
450 GOTO 660 !229
460 MN=4 !086
470 MX=30 !144
480 GOTO 660 !229
490 MN=5 !087
500 MX=31 !145
510 GOTO 660 !229
520 MN=6 !088
530 MX=31 !145
540 GOTO 660 !229
550 MN=7 !089
560 MX=30 !144
570 GOTO 660 !229
580 MN=8 !090
590 MX=31 !145
600 GOTO 660 !229
610 MN=9 !091
620 MX=30 !144
630 GOTO 660 !229
640 MN=10 !132
650 MX=31 !145
660 REM DUMMY STMT, START WI
TH 1ST DAY OF THE MONTH !054
670 DY=1 !085
680 GOSUB 900 ! DETERMINE TH
E DAY OF THE WEEK !024
690 FOR K=0 TO 5 !061
700 FOR I=D TO 6 !135
710 IF DY=0 THEN 790 !106
720 IF DY>MX THEN 780 !014
730 DOTW$(MONTH-1,K,I)=STR$(
DY)!174
740 IF DY<10 THEN DOTW$(MONT
H-1,K,I)=" "&DOTW$(MONTH-1,K
,I)!047
750 DOTW$(MONTH-1,K,I)=" "&D
OTW$(MONTH-1,K,I)!116
760 DY=DY+1 !179
770 GOTO 790 !104
780 DY=0 !084
790 NEXT I !223
800 D=0 !251
810 NEXT K !225
820 NEXT MONTH !028
830 FOR I=1 TO COPYNO :: GOS
UB 1300 :: NEXT I !098
840 DISPLAY AT(24,1):"PRESS
ANY KEY TO GO ON" !066
841 CALL SOUND(200,1397,5)!1
93
842 CALL KEY(0,K19,S19)!143
843 IF S19=0 THEN 842 !190
844 CALL CLEAR !209
845 DISPLAY AT(2,1):"PRESS 1
 - SHOW MONTH" !105
846 DISPLAY AT(4,1):"       2
 - PRINT MONTH" !204
847 DISPLAY AT(6,1):"       3
 - CHANGE PRINTER" !137
```

(See Page 36)

CALENDAR—

(Continued from Page 35)

```
848 DISPLAY AT(8,1):"        4
 - PRINT YEAR" !124
849 DISPLAY AT(10,1):"
5 - EXIT PROGRAM" !061
850 CALL SOUND(200,1397,5)!1
93
851 CALL KEY(0,K19,S19)!143
852 IF S19=0 THEN 851 !199
853 IF K19<49 OR K19>53 THEN
 850 !094
854 CALL CLEAR !209
860 ON K19-48 GOTO 882,884,8
69,870,890 !193
869 PFLG$="Y" !044
870 GOSUB 1280 ! CLEAR TABLE
 FOR NEXT CALENDAR !167
880 GOTO 102 !181
882 RUN "DSK1.CALENDAR1" !04
2
884 RUN "DSK1.CALENDAR2" !04
3
890 STOP :: CALL CLEAR :: GO
TO 890 !057
900 REM SUBR TO DETERMINE TH
E DAY OF THE WEEK (D) !044
910 REM SUN=0, MON=1,...,SAT
=6 !254
920 D=1+INT(2.6*MN-.2)+INT(Y
R/4+YR)+(INT(CN/4)-2*CN)!070
930 REM IF D<0, KEEP ADDING
7 TILL DAY (D) IS POSITIVE !
255
940 IF D>=0 THEN 980 !143
950 D=D+7 !007
960 GOTO 940 !254
970 REM MAKE MAKE D POSITIVE
 INTEGER !059
980 D=INT(7*INVERT(D/7)+.5)!
060
990 REM ADJUST D IF LEAP YEA
R REQUIRES IT !120
1000 IF YR+1<>0 THEN 1050 !2
49
1010 IF INVERT(CN/4)=0 THEN
1050 !105
1020 IF MN$="J" THEN 1040 !
LEAP YEAR TEST !022
1030 IF MN$<>"F" THEN 1050 !
 LEAP YEAR TEST !221
1040 D=D+1 !001
1050 RETURN !136
1060 REM SUBR. LOAD PRINTING
 CONSTANTS !204
1070 SP2$="  " !192
1080 SP3$="    " !226
```

```
1989                    CALENDAR                    1989

      JANUARY                 FEBRUARY                 MARCH

 S  M  T  W TH  F  S     S  M  T  W TH  F  S     S  M  T  W TH  F  S
                                    1  2  3  4                    1  2  3  4
 1  2  3  4  5  6  7     5  6  7  8  9 10 11     5  6  7  8  9 10 11
 8  9 10 11 12 13 14    12 13 14 15 16 17 18    12 13 14 15 16 17 18
15 16 17 18 19 20 21    19 20 21 22 23 24 25    19 20 21 22 23 24 25
22 23 24 25 26 27 28    26 27 28                26 27 28 29 30 31
29 30 31

      APRIL                   MAY                     JUNE

 S  M  T  W TH  F  S     S  M  T  W TH  F  S     S  M  T  W TH  F  S
                   1                    1  2  3  4  5  6                 1  2  3
 2  3  4  5  6  7  8     7  8  9 10 11 12 13     4  5  6  7  8  9 10
 9 10 11 12 13 14 15    14 15 16 17 18 19 20    11 12 13 14 15 16 17
16 17 18 19 20 21 22    21 22 23 24 25 26 27    18 19 20 21 22 23 24
23 24 25 26 27 28 29    28 29 30 31             25 26 27 28 29 30
30

      JULY                    AUGUST                  SEPTEMBER

 S  M  T  W TH  F  S     S  M  T  W TH  F  S     S  M  T  W TH  F  S
                   1                 1  2  3  4  5                    1  2
 2  3  4  5  6  7  8     6  7  8  9 10 11 12     3  4  5  6  7  8  9
 9 10 11 12 13 14 15    13 14 15 16 17 18 19    10 11 12 13 14 15 16
16 17 18 19 20 21 22    20 21 22 23 24 25 26    17 18 19 20 21 22 23
23 24 25 26 27 28 29    27 28 29 30 31          24 25 26 27 28 29 30
30 31

      OCTOBER                 NOVEMBER                DECEMBER

 S  M  T  W TH  F  S     S  M  T  W TH  F  S     S  M  T  W TH  F  S
                                    1  2  3  4                    1  2
 1  2  3  4  5  6  7     5  6  7  8  9 10 11     3  4  5  6  7  8  9
 8  9 10 11 12 13 14    12 13 14 15 16 17 18    10 11 12 13 14 15 16
15 16 17 18 19 20 21    19 20 21 22 23 24 25    17 18 19 20 21 22 23
22 23 24 25 26 27 28    26 27 28 29 30          24 25 26 27 28 29 30
29 30 31                                        31
```

Calendar printout reduced from original

```
1090 SP6$=SP3$&SP3$ !103
1100 SP9$=SP6$&SP3$ !109
1110 SP10$=SP9$&" " !134
1120 SP11$=SP10$&" " !175
1130 SP19$=SP10$&SP9$ !207
1140 SP20$=SP19$&" " !184
1150 SP21$=SP20$&" " !177
1160 SP22$=SP21$&" " !179
1170 SP23$=SP22$&" " !181
1180 SP24$=SP23$&" " !183
1190 SP27$=SP24$&SP3$ !205
1200 HEAD$(0)=SP9$&"JANUARY"
&SP20$&"FEBRUARY"&SP20$&"MAR
CH" !146
1210 HEAD$(1)=SP10$&"APRIL"&
SP23$&"MAY"&SP24$&"JUNE" !10
2
1220 HEAD$(2)=SP11$&"JULY"&S
P22$&"AUGUST"&SP19$&"SEPTEMB
ER" !165
1230 HEAD$(3)=SP9$&"OCTOBER"
&SP20$&"NOVEMBER"&SP19$&"DEC
EMBER" !094
1240 HOLD$="  S  M  T  W TH
 F  S" !005
1250 HEAD$(4)=SP2$&HOLD$&SP6
$&HOLD$&SP6$&HOLD$ !202
1260 GOSUB 1280 !085
1270 RETURN !136
1280 FOR M=0 TO 11 :: FOR K=
0 TO 5 :: FOR I=0 TO 6 :: DO
TW$(M,K,I)=SP3$ :: NEXT I ::
 NEXT K :: NEXT M ! CLEAR TA
BLE FOR NEXT CALENDAR !181
1290 RETURN !136
1300 REM SUBR. TO PRINT CALE
NDAR !238
1310 OPEN #1:PRNTR$ !005
1320 PRINT #1:HEAD$(5)!233
1330 PRINT #1:" " !048
1340 FOR Z=0 TO 9 STEP 3 ! G
```

(See Page 40)

MICRO-Reviews

40-column utilities, a columnizer and lotsa useful little programs

Ratings for the software reviewed in this column will be based on a star system as follows:

★ Leave it alone, back to the drawing board.

★★ Needs improvements, but workable.

★★★ A good program, worth trying.

★★★★ Send your money and buy it.

★★★★ 40-COLUMN UTILITIES

I don't know how many Extended BASIC programmers there are in the TI community. I would like to think a great many, but like myself, they find that most of the holes that needed filling in the software libraries, have been closed up. That's all well and good, but there is also a need to upgrade what you have done in the past.

This software will take you a long way toward accomplishing that goal. Be advised, this is NOT an enhanced graphics package, it's an enhanced TEXT package. For that reason, I like it a whole lot, and I expect to use it.

The most unique feature of this series of CALL LINKs is that it uses a couple of K of your stack for an additional screen. This extra screen can be used for anything and there is a nice demo program on the disk to give you an idea of what it's capable of. By pressing a single key, the demo goes from a 40-column text screen, to a graphics screen with moving sprites. Since the screen is already in VDP, there is no waiting. It's just there as soon as the key is pressed, and then back again on another press. A super idea for help files, games or whatever else you can dream up.

The rest of the LINKs include windowing, colors, scrolls within the windows, and a 255 character ACCEPT routine. There is also a CALL LINK("VERT",,,) and a CALL LINK("HORZ",,,) for 40-column graphics.

There are a few other routines of lesser importance, such as true lower case letters, but all have to do with text and strings.

The author also included a mergable set of SUBs to make your programming easier. For instance, CALL LINK("TEXT",A,B) can be used as CALL T(A,B). That saves a lot of programming bytes if you use them like that. By the way, the SUB is actually CALL TEXT, but you can shorten that to "T" if you want to.

The $5 the author is asking is a pittance for what these routines can do for you. They are well done, don't reduce the use of your regular commands, and don't require much room. Go ahead and buy it, you'll find plenty of uses for it.

Send $5 to: Brad Snyder, 148 Ave. 'A', Palmerton PA, 18071

★★★ CALCULATIONS, CONVERSIONS AND LOTSA DATA

That's an interesting title, but it might better be called "The Back of the Dictionary". If your dictionary weighs ten pounds or more you might well want to look into this disk.

CC&LD is a disk full of little programs that answer all those daily questions like what is the volume of a cone, how many teaspoons to a cup, the amount of the payment on the new Rolls, etc. Most of the items are fairly common, but there are some that go a step further, particularly in the data section. How about Prescription Latin Translation, Foreign Phrases, and the one I liked best, titled "Shop Stuff." This listing included:

1. Common nails—size, length, gage number, and quantity per pound

2. Finishing nails—same as above

3.Standard pipe—sizes, lengths, etc.

4. Glass—thickness, use, and maximum available size.

Although the disk smacks of an earlier commercial product, I thought it was quite well done, with something for everyone. Also, I found that each subject was very thorough, more so than the back of MY dictionary.

The information is handled through a series of menus that are quite user friendly. From the main menu you would select "Conversions", then "Household Conversions" from a secondary list. This brings up four screens full of various tables for fluid and solid quantity conversions. For instance, 1 Pint= 16 fluid ounces, 128 fluid drams, 473 milliliters, 96 teaspoons, 32 tablespoons, or, 2 cups. Very complete!

My only complaint was that I thought the programming could have been done a little more efficiently where the calculations were concerned. The data programs are quite large and demand separate files, not so the calculations. This is a minor detail though, because if you need this, you've got it in a very workable program. I would like to see the author categorize his subject matter on single disks, and expand on each one.

This suggestion will probably be taken up, too, because he freely admits he is crazy about "information software". Well done Rod! Send $5 to: Rodney Liewellyn, 107 August Dr., Seaford VA, 23696.

★★★★ NEW COLUMNIZER

As a newsletter editor, I was really impressed with this program. There have been a lot of attempts at creating text columns from TI-Writer files. Some have been successful, some not, simply because there is so much preliminary work, it wasn't worth it time wise. The best I had seen up to now only worked for two columns and was a bit slow.

This one prints out two, three or four columns, and has almost no preliminary work involved. The only restriction is that you must set you column width to 33 or 27, depending on the format you need. The text file is saved in the normal manner via SF(Save File) and that's all there is to it.

(See Page 40)

Form Shop

Flexible, quality forms from your TI and it's easy to use

By BILL GASKILL

The thousands of 99ers who have remained active in the TI community will surely be celebrating this year. As we near the sixth anniversary of our abandonment by Texas Instruments, the hardware and software that is available for the 99/4A continues to prove what can be done with our computer by its loyal users.

One of the first entries out of the chute for 1989 is Form Shop, which was formally introduced at Fest-West '89 in San Diego. Form Shop is the first of its type in the TI community and it is reasonably priced at $15. It delivers what it promises, and it is easy to use.

Form Shop is, as the name suggests, a forms design program, forms being virtually anything that you can think of that uses borders, boxes and intersects created with horizontally or vertically placed single-line, double-line or thick-line drawings. The program gives you all the necessary tools to create computer-quality forms for business or personal use, and it is all done in the familiar TI-Writer environment. So for most users, the learning curve will be minimal.

Form Shop requires a memory expansion, Extended BASIC, a disk drive and a printer that supports the IBM character set. This would include Epson or Epson-compatible printers, the Gemini 10X, NX10, NX1000, Citizen MSP20 and the Panasonic 10911. Form Shop cannot be used with the Prowriter and Axiom printers. It can apparently be used with the Gemini 10 printer, but the results are poor. There are no doubt a host of other printers that can be used with Form Shop. The best way to tell is to check the printer manual to determine if it is Epson-compatible or if it supports the IBM graphics character set.

The Form Shop product consists of a four-page manual, one SS/SD disk (which is not copy-protected), a custom-designed loader for Extended BASIC, a TI-Writer-like editor and formatter, several support files designed to allow you to adapt the system to your printer, three sample forms, three "Test" files that allow the sample forms to be printed out in succession and an UNSET transliterate file designed to reset the Control codes sent to your printer.

If you want to get an idea of what you can do with Form Shop, just look at the title screen when loading the program. The title screen lists such applications as bar graphs, (hand) bills, charts, graph paper, ledgers, maps, organizational charts, questionnaires, receipts and tests.

After loading the title screen, press the spacebar and a two-option menu appears: Press 1 to create a form or 2 to print a form. TI-Writer users will be pleased to note that from this point on most of what is needed to use the program varies only slightly from the operation of TI-Writer. This is a very strong point in Form Shop's favor, since most computer users balk at having to learn programs with radically different user interface schemes.

In looking through the files on the Form Shop disk I discovered that the SAMPLE3 file is a diagram of the 99/4A keyboard, set up so that you can see which keys draw each particular type of line or intersect. I would strongly suggest printing this file first. Before doing so, I had to load SAMPLE3 into the editor (Create a Form) and insert a .IF DSK1.SETUPALL statement in line 1 of the file. I use an Epson EX-800 printer. I then had to tack a .CR onto the end of the PIO.LF printer name that comes as the default in the formatter.

While the keyboard printout is nice, Rodger Merritt, the author, has also included a legend of the keypresses in the editor program. The only problem with it is that it appears on the same line at the bottom

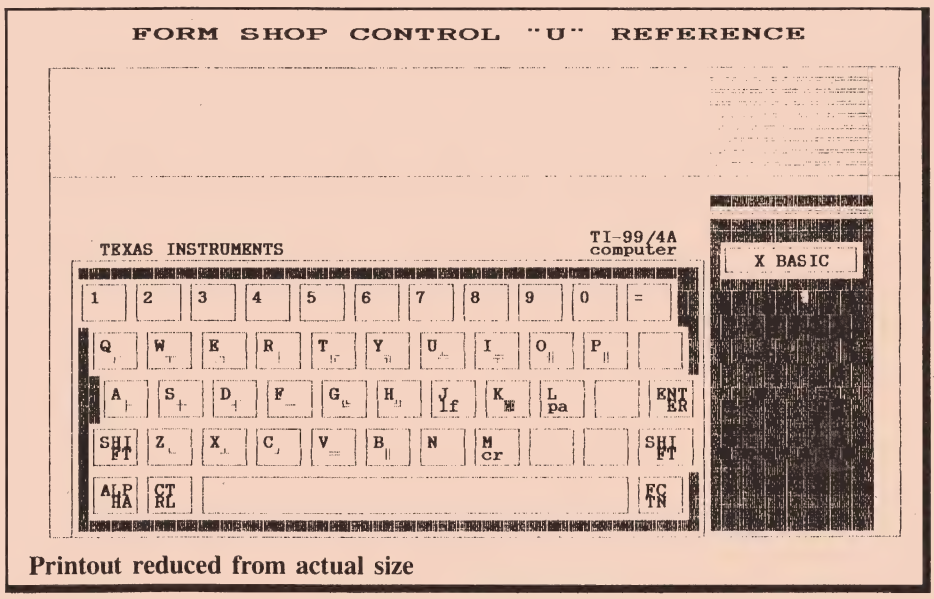
(See Page 39)

Review

Report Card

Performance . . . . . A
Ease of Use . . . . . A–
Documentation C
Value . . . . . A+
Final Grade . . . . . A

Cost: $15 + $2 S/H
Distributor: Comprodine, 1949 Evergreen Ave., Fullerton, CA 92635
Requirements: Console, XB, memory expansion, disk drive, printer



Printout reduced from actual size

FORM SHOP—

(Continued from Page 38)

of a file that Texas Instruments put the "End of File V1.0" message on. This means that the legend disappears from the screen when your form is longer than 24 lines. So the printout really seems to be the best resource, unless of course you don't mind paging up and down to look at the legend. However, I found the program so simple and easy to use that I had the most common keypresses (there are only about 24 of them) memorized in an hour or so of working with the program.

Creating a form is simply a matter of pressing Ctrl U to enter the transliterate or "Control U" mode and then pressing the keys that display the kind of lines or intersects you want to use. I found that I needed to put line feeds at the right margin of all of the forms that I designed and each file printed out properly only when I included the instruction .IF DSK1.SETUPALL at the beginning of the file. Also, as I mentioned earlier, the .CR had to be added to my printer device name in the formatter.

When compared to other forms generation programs that I have used on IBM-type computers, Form Shop stacks up very well. The fact that it is working on a TI-99/4A has no bearing on anything as far as speed. A form created with Form Shop prints in about 3-4 minutes, depending upon the mixture of text/graphics characters, the size of the form and, of course, the capabilities of the printer. The program that I use at work that runs on an IBM-type computer prints no faster.

What Form Shop does not have, that is supported by the IBM-type forms designer program, is the ability to design a form that is more than 80 columns wide. Alas, because the author of TI-Writer limited the page width to 80 columns, Form Shop is also limited to 80 columns for What You See Is What You Get (WYSIWYG) type forms. It may be possible to design forms in 80-column mode and have them print out in more than 80 columns, just as one can do with text files, but I haven't looked into that possibility.

OUTPUT

As you can see from the sample forms in Fig. 1, the printed product generated by Form Shop is terrific. Although one's printer will play an important part in the end result of any form, Form Shop does a nice job of sending the "right stuff" to my printer to get the quality that I need.

JANUARY 1989

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
1 HAPPY NEW YEARS	2	3	4	5	6	7
8 Bash at Freddy's 666 Elm St	9	10	11	12 Fly to Kokomo with the Cruise	13 Stay in bed ALL DAY	14
15	16 5 AM hunting with Sly bring M60	17	18	19	20	21
22 Quayle hunting with the Prez	23	24	25	26	27 Go with Don King for a haircut	28
29	30	31				

Printout reduced from actual size

The various "lines" that FORMSHOP is capable of producing include single-line drawings (Fig. 1 Playoffs illustration); double-line drawings (Fig. 1 "Champion" box, Playoffs illustration); and thick line drawings (Fig. 1, horizontal bars, Expense Graph illustration). Form Shop does not do circles, elipses, thin lines or hatched lines. My IBM-type forms designer also doesn't do circles or elipses, but it does offer thin lines and hatched lines.

HARDWARE

Form Shop runs on both the 99/4A and the Myarc Geneve 9640. It can be ported over to BA-Writer, My-Word, FunnelWeb and Art Green's TI-ReWrite V4.2. It can be loaded from a mechanical floppy drive and from the Horizon Ram Disk. I have tested both. I have not tested it on the Cor-Comp, Grand RAM, Myarc or Rave 99 RAMdisks. However, Form Shop will not load from a hard drive. I use Form Shop on my Horizon RAMdisk (HRD), and it loads in about 5 seconds. If you are using it from a mechanical drive you can speed up the load time by doing away with the pathing through the title screen. On your backup copy, rename the load file as LOADX and then rename the FSLOAD file as LOAD. You won't see the attractive title screen anymore, but you won't have to sit around waiting for the program to get up and running either.

On my system, which uses John Johnson's Menu with RAMdisk Operating System V7.35 on the Horizon RAMdisk, I put Form Shop on as a menu choice and copied the program files to DSK4 (the second half of my HRD). I have the FSLOAD file called directly from the menu and I modified the DSK1.FSMENU statement in line 10 of the FSLOAD file so that it reads DSK4.FSMENU. I love those lightning fast loads.

PRINTER SUPPORT

Merritt has done a good job of providing support for the most common printers. While owners of the C. Itoh Prowriter and the Axiom line of printers will probably argue that their printers are "common," too, I can only say that for $15 the Form Shop program supports a broad spectrum of printers, and hits the majority of printer owners dead-center.

DOCUMENTATION

If Form Shop has a weakness it is in the

(See Page 40)

FORM SHOP—

(Continued from Page 39)

instructions. The manual that comes with Form Shop is no doubt designed to be attractive to the eye for marketing purposes, and it does look good. The problem is that appearance isn't much of a substitute for information when you are trying to figure a program out. Because my copy of Form Shop was purchased at Fest-West '89, and my informant at the faire told me that the packaging was hurriedly put together the night before the faire so that it would be ready for the show, the docs that I have may be a little different than the ones now sold with the product. I hope so. Even though the program is easy to use, there are some things that a first-time user needs to know that are not adequately covered in the manual. Much of the information about line feeds, carriage returns, SETUP files and the like was discovered by trial and error. A user really shouldn't have to waste the time nor the printer paper learning the program in this way. The instructions should tell me how to use the program.

CONCLUSION

Form Shop is a neat product. I have found many uses for it and will no doubt find many more. While I have the same type of application available for my IBM clone (at a cost of $89.95), there is nothing like having it for my TI! If you have a need to create custom forms for whatever use, then Form Shop is the tool for you.

MICRO-REVIEWS—

(Continued from Page 37)

Once your text file is complete, you boot up the columnizer. There are five questions to answer before the program takes off:

1. Type of printer in use (Epson, Gemini 10X, or, Panasonic)
2. Number of Columns (2,3,4)
3. Right Justify (Y/N)
4. Print a Title (Y/N)
5. Print Quality (Draft or Bold)

For two columns, you get regular Pica print, for three you get Elite, and for four columns it prints in condensed. In will right justify all columns in any mode of operation if you wish. And one very nice feature cleans up a problem that has always bugged the heck out of me. Let's say you end a paragraph and the last line has two words on it. Many previous columnizers, while justifying, put one word on the left and the other on the far right. YUK! With this one if you put a plus (+) sign at the beginning of the line it leaves it alone. There are obviously other reasons to do this also, like numeric columns, etc.

Another nice feature is the titling. If you elect for a title, it will print up to two, 28-character title lines, centered on the first page in double wide text. SUPER!

Also unlike some columnizers, you are allowed to indent via leading spaces. (I hate leaving blank lines to separate paragraphs.)

Finally, you have the convenience of a quick draft dump, or a final copy in Double Strike/Bold mode. Who could ask for more.

The program is done in Extended BASIC with CALL LINKs to speed everything along. Also it should be noted that it doesn't care how long the file is, it just keeps printing page after page.

If you are like me and do everything in 40-column mode, there is a program included that automatically resets the margins for you. A very handy utility for a lot of things.

If it sounds like I'm raving about this one, well, maybe I am, but anything that will help in the writing task of newsletters etc., always gets my vote. This is the best of a kind so far.

Send $5 plus $1.50 for disk and postage to: Ron Prewitt, 6429 South Fife, Tacoma WA, 98409.

A couple of comments to end with this month.

First, this column may be a little short or even non-existent this summer because of two things: I am writing a novel and that is taking a lot of my time and, also, I expect the in-coming software to slow down. If it doesn't, no problem. I do expect to do a major review of Page-Pro in the next month or two, however, so you can look forward to that.

Concerning your software submissions; I have heard a few comments about the fact that nobody ever receives less than three stars. So for all you sports fans that go to the game looking for blood, here's the reason for that: If a fairware author sends me a program for review, I look at it, and if it seems like it's going to require less than three stars, I send it back to him, with comment. The reason is, that I see no point in cutting a person down because he is just starting out. A lot of the new authors need help, not the whipping post. On the other hand, if it's a commercial piece of software, (a company), they are going to get reviewed no matter what the rating. These guys have been around long enough to know the community's needs, and what we'll put up with.

I have also been accused of being opinionated ... seen any good movies lately? What was your opinion of them, Mr. Reviewer?

Nuff said ... Have a marvelous summer.

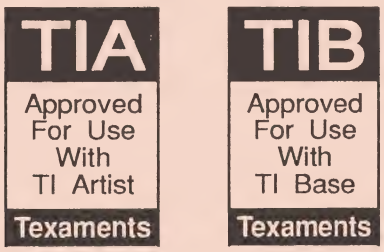
If you would like me to review your software in this column, please send it to the address below, and if you would like it returned, include a SASE. Write to: Harry T. Brashear, 2753 Main St., Newfane, NY 14108.

CALENDAR—

(Continued from Page 36)

```
ROUP OF 3 MONTHS !Ø47
135Ø PRINT #1:HEAD$(Z/3)!ØØ5
136Ø PRINT #1:" " !Ø48
137Ø PRINT #1:HEAD$(4)!232
138Ø PRINT #1:" " !Ø48
139Ø FOR Y=Ø TO 5 ! LINE !Ø2
2
14ØØ HOLD$=SP2$ !ØØ2
141Ø FOR Q=Ø TO 2 ! MONTH IN
 LINE !1Ø4
142Ø FOR X=Ø TO 6 ! DAY OF T
HE WEEK IN LINE !2Ø5
143Ø HOLD$=HOLD$&DOTW$(Z+Q,Y
,X)!Ø94
144Ø NEXT X !238
145Ø IF Q=2 THEN 147Ø !2Ø2
146Ø HOLD$=HOLD$&SP6$ !ØØ9
147Ø NEXT Q !231
148Ø PRINT #1:HOLD$ !147
15ØØ NEXT Y !239
15Ø5 PRINT #1:" " !Ø48
151Ø NEXT Z !24Ø
1515 PRINT #1:CHR$(12)!184
152Ø CLOSE #1 !151
153Ø RETURN !136
154Ø END !139
```

Newsbytes



Texaments announces certification program

Texaments has announced an official compatibility program for all TI-Artist and TI Base add-on software.

Steve Lamberti, president of Texaments, says any software, whether commercial or freeware, developed exclusively for use with the TI-Artist or TI Base or in conjunction with either package is covered by the program. Examples of such software include Artist's Companion #1 (fonts and clipart for TI Artist), MICROdex for TI Base (information databases and templates) and Designer Labels (standalone utilities), he says.

Lamberti says some add-on software is "simply not compatible with its target application programs." He says the "Approved" emblems will be issued only to software authors and publishers for software Texaments has tested and approved for compatibility. To obtain certification and the right to use the emblems, authors and publishers must provide Texaments with a complete working copy of their software, Lamberti says. On approval, a letter of certification along with camera-ready copies of the "Approved" emblems will be awarded to the author or publisher, he says.

He says certification is free and takes approximately four to six weeks. For certification, send software to Certification Department, Texaments, 53 Center St., Patchogue, NY 11772.

Lamberti says Texaments is attempting to compile and organize articles on TI-Base which have appeared in various publications and to make them available free of charge.

He asks anyone who has written an article, a command (program) file or a utility of any kind to send them to Dept. TIBC at Texaments at the address above. Submissions should be on floppy disks when possible.

Submissions will be placed in a special TI Base Forum on TI Source, a free multiuser bulletin board system sponsored by Texaments. Submissions may also be directly uploaded by modem to TI Source at (516) 475-6463 24 hours a day.

Name address and telephone number should be included with any submissions or correspondence.

Lamberti says Texaments is beta testing a new high-speed sorting utility for TI Base. This is designed as a "standalone program" that will sort database files three to four times faster than the sort directive now within TI BASE. Price and availability will be announced during the third quarter of 1989, he says.

For further information, write Texaments at the address above or call (516) 475-3480.

Group selling disks of adventures for TI

The Massaschusetts Users of the Ninety-Nine Computer and Hobbists group (M.U.N.C.H.) is selling the TI Adventure Compendium disk set, which it describes as all the fairware and public domain adventures ever written for TI99/4A and Geneve computers.

All game files are archived. Barry Boone's Archiver program and the *README file are also included.

The copyrighted compilation is a joint fund-raising venture of M.U.N.C.H. and the Western New York 99ers. The listing was compiled with the help of Mickey Schmitt's *Adventure Reference Guide*, available from Asgard Software.

Several of the games require an Adventure or Tunnels of Doom module. Otherwise, they will work with BASIC, Extended BASIC, Mini-Memory, Terminal Emulator II or Editor/Assembler.

Price is $11.95 for three double-sided, double-density disks; $15.95 for five double-sided, single-density disks; or $19.95 for 10 single-sided, single-density disks.

To order, make checks out to M.U.N.C.H. and mail to the group at P.O. Box 7193, Worcester, MA 001605-7193.

James W. Cox, treasurer and editor of M.U.N.C.H., reports that Sughrue was recently in a car crash that involved numerous head injuries, but that orders for Sughrue's fairware PLUS! program will be filled in fairly short order.

Cox says Sughrue is unable to receive telephone calls, but wants to thank 99ers around the country for their cards and letters.

Chicago TI-Faire set

The Chicago Area TI99/4A Users Group has scheduled its annual TI-Faire for Nov. 4 at the Holiday Inn at 3505 Algonquin Rd. in Rolling Meadows, Illinois.

A social mixer will be held from 8 p.m. until midnight Nov. 3. Admission charge is $5. The Faire will be held from 9 a.m to 5 p.m. the following day. Faire admission is $4. A dinner will be held at 7 p.m. after the Faire. Dinner tickets are $15 each.

Exhibit space is $75 per eight-foot table. Hotel room rate is $50 for a single, double or tower room. Limousine service is free to and from O'Hare Airport and the hotel.

The TI-Faire is held in conjunction with the Milwaukee TI-Faire Nov. 5 at the Quality Inn, 5311 S. Howell Ave., Milwaukee, Wisconsin (across from Mitchell Field Airport).

For further information on the Chicago TI-Faire, contact Sandy Bartels, Chicago TI99/4A Users Group, P.O. Box 578341, Chicago, IL 60657 or phone (312) 859-3850. For information on the Milwaukee TI-Faire, contact Gene Hitz, 4122 N. Glenway, Wauwatosa, WI 53222 or (414) 535-0133.

TI 'World's Faire' set

Billed by its sponsoring group as the "World's Faire of TI Shows," TI International Expo 89 is scheduled for Sept. 16 at the Howard Johnson Inn in Alexandria, Virginia.

The event is sponsored by the Mid-Atlantic Ninety-Niners (MANNERS).

The show is scheduled for 9 a.m.-5 p.m. on the exhibition floor at the inn, at 5821 Richmond Highway (Route 1 South and I-95 on the Capital Beltway — Exit 1S). A preshow reception will be held at 6 p.m. the night before the event and a postshow reception at 6 p.m. on the event date, fol-

(See Page 42)

Newsbytes

(Continued from Page 41)

lowed by a buffet dinner at 7 p.m.

Show admission is $3 in advance, $4 at the door. Advance buffet dinner tickets are $15 for adults, $7 for children to age 11.

Booths are available to commercial vendors for $50 for the first table and $15 for the second table. Two free show admissions are included. Electricity is available but limited. User groups should contact Expo 89 for booth rates.

Discounted show rate for the hotel is $55 plus tax per night. Reservations should be made by calling the hotel at (703) 329-1400 and mentioning Expo 89. International attendees may write Expo 89 for room reservations.

Besides booths, the event will feature hardware and software workshops, product demonstrations, panel discussions and speakers, according to organizers.

For further information contact MANNERS, TI International Expo 89, P.O. Box 40005, Rockville, MD 20850 or (301) 340-7179. Information can also be obtained by contacting TELEDATA on the Delphi TI NET or 74405,1207 on CompuServe.

TI-Fest West '90 set

TI-Fest West '90 is scheduled for Feb. 17-18 at Day's Inn in Tucson, Arizona, 88 E. Broadway, just off I-10, sponsored by the Southwest 99ers.

Ida McCargar of the Southwest 99ers says room reservations may be made by calling Day's Inn at (602) 791-7581 and mentioning Fest-West for special rates of $49.18 single or double (including taxes). Reservations must be made by Jan. 16; first night deposit or Visa or MasterCard number required.

For information, call (602) 747-5046. Information is also posted on GEnie and on the Cactus Patch BBS, (602) 795-1953.

Boone offers EPROM for Mechatronics card

Barry Boone is offering a new upgrade EPROM for Mechatronics 80-column cards. According to Boone, the EPROM fixes all the bugs in the EPROM that was shipped with the unit, such as the improper operation of the OPEN #1:"TEXT80" command, and has many new and useful features, including:

- The 80-column card may be set at any CRU base desired. The old EPROM required the use of CRU>1000, which conflicted with RAMdisks and other devices.
- The interlace mode may be selected using a DIP switch.
- All DIP switches are now read properly. The old EPROM would not read the switches when in the TEXT80 mode, and the result was improper operation (technically speaking, it put the card in PAL mode, which worked only in Europe).
- The mouse port is automatically enabled on powerup. The old EPROM left the mouse port turned off, and users had to know assembly language to turn the mouse port on. (The Geneve and the DIJIT 80-column card automatically turn it on, and all software that uses a mouse assumes that it is already turned on).
- Extensive documentation on using the card to its fullest, including information on how to attach devices such as a mouse or lightpen (with detailed pinouts of the I/O ports on the back of the unit).
- A disk containing software for the 80-column card, including versions of ROS 7.3 for the Horizon RAMDISK, which Boone modified to work with the 80-column card, and instructions on determining which ROS to use. Other public domain software includes programs to view MY-Art pictures (in up to 256 colors), an external DSR file for TI-Artist so a mouse may be used, a version of TI-Writer that works in 80 columns (unlike the one shipped with the unit), and a list of other software available for the 80-column card.

The EPROM is available for $20 (U.S.), plus $2 shipping and handling. It is available from: Barry Boone, P.O. Box 1233, Sand Springs, OK 74063. Phone: (918)356-4648, 8-10 p.m. CST weekdays, 10 a.m.-10 p.m. CST weekends.

Any updates to the EPROM will be available for $5, including shipping, provided that the buyer returns a good EPROM for the update.

Boone is also offering to install any Editor/Assembler Option 5 type program into the EPROM, providing it meets the following criteria: It must be one file, 33 sectors or less, and must be compatible with any standard assembly program image loader. For example, Archiver III could be burned into the EPROM, and it would be available to users from anywhere by issuing either a CALL ARC command, or by accessing it like a file (e.g. DELETE "ARC" or OLD ARC would also load it.) The charge for this service is an additional $10 to cover time and the more expensive EPROM required. The $10 will be refunded if the program is incompatible. The buyer must provide the program when the order is placed.

Also, according to how much of a demand there is, Boone is considering a version of the EPROM that will turn the 80-column card into a print spooler. The price/specifications are as yet undetermined for this version.

***Newsbytes* is a column of general information for TI and Geneve users. Send items to MICROpendium Newsbytes, P.O. Box 1343, Round Rock, TX 78680.**

XBASIC—

(Continued from Page 14)

```
29325 DIM T(16)!124
29330 C$=C$&RPT$("0",16)!110
29335 FOR L=1 TO 16 :: T(L)=
ASC(SEG$(C$,L,1))-48 !088
29340 IF T(L)>9 THEN T(L)=T(
L)-7 !157
29345 NEXT L !226
29350 FOR L=1 TO 8 :: C(L)=0
 :: NEXT L !219
29355 FOR L=1 TO 2 :: FOR L2
=L TO 16 STEP 2 :: FOR P=0 T
O 3 !035
29360 IF (T(L2)AND 2^P)=2^P
THEN C((L-1)*4+4-P)=C((L-1)*
4+4-P)+2^(INT((16-L2)/2))!10
4
29365 NEXT P :: NEXT L2 :: N
EXT L !224
29370 T$=CHR$(27)&CHR$(75)&C
HR$(8)&CHR$(0):: FOR L=1 TO
8 :: T$=T$&CHR$(C(L)):: NEXT
 L !028
29375 SUBEND !168
```

User Notes

One step further with clock program

This comes from Harold Hoyt Jr., of Spanish Lake, Missouri. He writes:

The clock program, as originally written, by D.L. Fitchhorn, and published in the April 1989 MICROpendium User Notes, converts hex data to base 10 data and uses CALL LOAD in Extended BASIC to put the converted data into memory. You may substitute any hex data strings representing a *Memory Image* program that is RUNnable from XBASIC for the DATA statements in this program, which makes it a dandy utility.

I have carried this idea a step further by reworking Fitchhorn's program, so that it will write the CALL LOAD commands to a text file, which can then be converted to a program, which may be saved and run. Some of the results of my efforts were published in the January 1989 St. Louis UG Newsletter.

The extra steps required to create this new program are worthwhile, since the steps are done only once to create the new program, and the resulting new program will run much faster. Fitchhorn's original program took 61 seconds to run, due mostly to the time required to convert all the hex data to CALL LOADs. The program HCLOCK derived by my method runs in 4.5 seconds.

Two methods have been traditionally used to make programs write other programs. One method opens a file in MERGE format, D/V 163, and sends BASIC line number data and Tokens to this file. The end program is then recovered by using a MERGE command to convert the D/V 163 file to an XBASIC program.

The second method opens a D/V 80 file and writes a program LISTing to the file. The LISTing is then converted to an XBASIC program using a utility that converts a LISTing to a program using a Token look-up table. I like the second approach better since it is not machine type dependent, and it allows one to convert any text file program LISTing from any machine to TI XBASIC.

Assume that you have some source code that could be assembled and run using XBASIC. This could be substituted for the data in Fitchhorn's clock program. I left the data and the framework of his program, deleting the check routine. Then, a a LISTing file was opened in line 110, and a SUBroutine was attached to write one CALL LOAD line of text, including the line number. Lines 1500-1600 do this. Line 120 sets up the line number for the LISTing, the decimal POKE address and the H$ used in the hex to decimal conversion. Lines 130 and 140 demonstrate the ability to POKE address lines to the LISTing directly. Lines 500 to 990 POKE the last lines to the LISTing, including the look-up of the "LFAL" ("Last Free Address in Low Memory") etc.

If you have the source code of a program that will run in XBASIC and want to convert it to CALL LOAD format, the following is one way of doing it, using the modified program, renamed TICK/HH, as a utility.

To generate the hex data for the program, use the Editor/Assembler cartridge and E/A disk, (not Funnelweb) to edit the source code and place an AORG statement at the beginning of the source code. The reason I want to use the E/A cartridge version of the assembler is to create a LISTing that shows the source and object code. The AORG statement forces the assembler LISTing to show non-relocatable code in the LISTing. The hex data in the LISTing may be edited by TI-Writer and APPENDed to the TICK/HH program LISTing as DATA. Ordinarily, the assembler will calculate the proper address for the relocatable code, but by using a CALL LOAD format, We take over this job, and need the AORG statement to force the actual addresses into the assembler LISTing.

We need to update the FFAM (First Free Address In Memory). Be sure to not use the "C" Compressed object code option as this is not usable from XBASIC. My favorite utility for converting LISTings to programs is Steve Karasek's SuperBasic (reviewed in the February 1989 issue of MICROpendium).

```
1 !SAVE DSK1.TICK/HH !253
10 !----------------------!
 !175
20 ! EXTENDED BASIC program!
 !175
30 ! based on D.L.Fitchhorn!
 !113
40 ! Interrupt driven clock!
 !101
50 !  By: H. C. Hoyt Jr.   !
 !046
60 !       1380 Trampe Ave  !
 !053
70 !       Spanish Lake, Mo.!
 !013
80 !63138 1/3/89,rev5/12/89!
 !220
90 !----------------------!
 !175
100 !Writes data to a CALL L
OAD program listing. You con
vert listing to XBasic and t
hen RUN it. !114
110 OPEN #1:"DSK1.LISTING",O
UTPUT !035
120 LN=500 :: R=1024Ø :: T=2
4 :: HH,MM=Ø :: HX$="Ø123456
789ABCDEF" !101
130 PRINT #1:"1 !SAVE DSK1.H
CLOCK" !Let's Keep track of
what program is where! !045
140 PRINT #1:STR$(LN);" CALL
 PEEK(8198,A):: IF A<>170 TH
EN CALL INIT" :: LN=LN+10 !1
92
150 GOSUB 1500 !049
160 IF X$<>"END" THEN 150 !0
47
500 !EXIT Routine !195
510 LN=LN+10 :: X$=STR$(LN)&
" CALL PEEK(8196,A,B) :: LFA
L=A*256+B  :: NEWL=LFAL-16 :
: A=INT(NEWL/256) :: B=NEWL-
A*256" !101
515 PRINT #1:X$ !196
520 LN=LN+10 :: PRINT #1:STR
$(LN);" CALL LOAD(8196,A,B):
: CALL LOAD(NEWL,83,84,65,82
,84,32,40,0)" !170
530 LN=LN+10 :: PRINT #1:STR
$(LN);" CALL LOAD(NEWL+8,83,
84,79,80,32,32,40,40):: CALL
 LINK(""START"")" !108
540 LN=LN+10 :: PRINT #1:STR
$(LN);" CALL LOAD(10543,T,0,
HR,0,MN,0,SC)" !212
990 CLOSE #1 :: END !164
1000 DATA C820,28EA,292C,C82
0,28E8,2928,C820,28E6 !224
```

(See Page 44)

User Notes

(Continued from Page 43)

```
1010 DATA 292E,04E0,2930,04E
0,2932,04E0,2934,0200 !119
1020 DATA 282E,C800,83C4,045
B,04E0,83C4,045B !104
1030 DATA 02E0,2928,02E0,292
8,0602,1652,C0A0,28EA !154
1040 DATA 0586,0286,003C,160
E,04C6,0585,0285,003C !129
1050 DATA 1609,04C5,0584,80C
4,1605,04C4,0283,0018,1301,0
584 !005
1060 DATA 06C0,D800,8C02,06C
0,E020,28E4,D800,8C02 !182
1070 DATA 4020,28E4,D064,28E
C,0941,0221,9000,D801,8C00 !
228
1080 DATA 0A41,0241,0F00,022
1,9000,D801,8C00,0201 !084
1090 DATA 9A00,D801,8C00,D06
5,28EC,0941,0221,9000 !151
1100 DATA D801,8C00,0A41,024
1,0F00,0221,9000,D801 !110
1110 DATA 8C00,0201,8C00,D80
1,8C00,D066,28EC,0941 !169
1120 DATA 0221,9000,D801,8C0
0,0A41,0241,0F00,0221 !086
1130 DATA 9000,D801,8C00,072
0,83D6 !138
1140 DATA 02E0,83E0,045B,400
0,000D,0017,003B !032
1150 DATA 0001,0203,0405,060
7,0809,1011 !138
1160 DATA 1213,1415,1617,181
9,2021,2223 !156
1170 DATA 2425,2627,2829,3031
,3233,3435 !174
1180 DATA 3637,3839,4041,424
3,4445,4647 !192
1190 DATA 4849,5051,5253,545
5,5657,5859,END !039
1500 !SUB WRITE LINE(LN,R,X$
,A,B) !076
1510 READ X$ !019
1520 IF X$="END" THEN 1600 !
030
1530 PRINT #1:STR$(LN);" CAL
L LOAD(";STR$(R);",";!254
1540 FOR I=1 TO 8 !063
1550 A=(POS(HX$,SEG$(X$,1,1)
,1)-1)*16+POS(HX$,SEG$(X$,2,
1),1)-1 :: B=(POS(HX$,SEG$(X
$,3,1),1)-1)*16+POS(HX$,SEG$
(X$,4,1),1)-1 !207
1560 PRINT #1:STR$(A);",";ST
R$(B);:: R=R+2 :: IF I=8 THE
N 1590 !206
1570 READ X$ :: IF X$="END"
THEN 1590 !169
1580 PRINT #1:",";:: NEXT I
!081
1590 PRINT #1:")" :: LN=LN+1
0 !153
1600 RETURN !136
```

```
1 !SAVE DSK1.HCLOCK !199
500 CALL PEEK(8198,A):: IF A
<>170 THEN CALL INIT !011
510 CALL LOAD(10240,200,32,4
0,234,41,44,200,32,40,232,41
,40,200,32,40,230)!252
520 CALL LOAD(10256,41,46,4,
224,41,48,4,224,41,50,4,224,
41,52,2,0)!139
530 CALL LOAD(10272,40,46,20
0,0,131,196,4,91,4,224,131,1
96,4,91,2,224)!093
540 CALL LOAD(10288,41,40,2,
224,41,40,6,2,22,82,192,160,
40,234,5,134)!038
550 CALL LOAD(10304,2,134,0,
60,22,14,4,198,5,133,2,133,0
,60,22,9)!079
560 CALL LOAD(10320,4,197,5,
132,128,196,22,5,4,196,2,131
,0,24,19,1)!200
570 CALL LOAD(10336,5,132,6,
192,216,0,140,2,6,192,224,32
,40,228,216,0)!087
580 CALL LOAD(10352,140,2,64
,32,40,228,208,100,40,236,9,
65,2,33,144,0)!085
590 CALL LOAD(10368,216,1,14
0,0,10,65,2,65,15,0,2,33,144
,0,216,1)!075
600 CALL LOAD(10384,140,0,2,
1,154,0,216,1,140,0,208,101,
40,236,9,65)!227
610 CALL LOAD(10400,2,33,144
,0,216,1,140,0,10,65,2,65,15
,0,2,33)!011
620 CALL LOAD(10416,144,0,21
6,1,140,0,2,1,140,0,216,1,14
0,0,208,102)!203
630 CALL LOAD(10432,40,236,9
,65,2,33,144,0,216,1,140,0,1
0,65,2,65)!135
640 CALL LOAD(10448,15,0,2,3
3,144,0,216,1,140,0,7,32,131
,214,2,224)!171
650 CALL LOAD(10464,131,224,
4,91,64,0,0,13,0,23,0,59,0,1
,2,3)!125
660 CALL LOAD(10480,4,5,6,7,
8,9,16,17,18,19,20,21,22,23,
24,25)!206
670 CALL LOAD(10496,32,33,34
,35,36,37,38,39,40,41,48,49,
50,51,52,53)!023
680 CALL LOAD(10512,54,55,56
,57,64,65,66,67,68,69,70,71,
72,73,80,81)!052
690 CALL LOAD(10528,82,83,84
,85,86,87,88,89)!023
710 CALL PEEK(8196,A,B):: LF
AL=A*256+B :: NEWL=LFAL-16 :
: A=INT(NEWL/256):: B=NEWL-A
*256 !227
720 CALL LOAD(8196,A,B):: CA
LL LOAD(NEWL,83,84,65,82,84,
32,40,0)!041
730 CALL LOAD(NEWL+8,83,84,7
9,80,32,32,40,40):: CALL LIN
K("START")!247
740 CALL LOAD(10543,T,0,HR,0
,MN,0,SC)!175
```

Cartridge port extender warning

This comes from Denver Earl Sullivan of Osgood, Indiana. He writes:

Many of us have seen those 11-inch cartridge port extended cables in TI-oriented computer catalogs. Beware: I purchased one of these so-called "wear reducers" and installed it as stated in the manual, and it burned up over 15 chips, resistors and other digital components in the computer console.

It is much cheaper and easier to purchase a GROM extension assembly from Texas Instruments (Part No. 1049693-0001, about $12) and install that when and if your cartridge port begins to short out. This is a worthwhile investment.

'Moving arrow' update

This comes from Russ Stanton, of Alexandria, Lousisiana. He writes:

To all who tried to make the "Moving Arrow" program in February's issue work,

(See Page 45)

User Notes

(Continued from Page 44)

and failed to do so, my humble apologies. There are a couple of things I left out of the article.

First, the program is not a database. It is a disk-based program. One file per entry only. In other words, once you have inputted name, address and phone number you save it to diskette. Then you start all over again inputting a new record. The program works that way because I use a much larger program with that type of disk file system for my carpet business.

Now that I've cleared up the type of filing system it has, let's get down to business on how the program really works.

1. To make the arrow up type in a "R" and press Enter.

2. To move the arrow down, enter your information and press Enter. You may also move the arrow down by simply typing Enter. The cursor will move down without disturbing the data on that line. That's the good thing about the input system in this program.

3. The program was mainly written to show:

- How to save memory space by using GOSUBs and not retyping ACCEPT AT and DISPLAY AT statements over and over.
- How to be able to go back and fix an input goof without having to start over.
- How to write a structured program with the use of pre-scan controls.
- How to use error trapping code in the TI99/4A.
- How to set up a simple file system.

4. The "RETURN TO MAIN DISK MENU" needs a file called "LOAD" to return to, otherwise you'll get a file error warning.

If you want to make a batch file out of the program, just create an array and you'll have it.

Anyone wishing to correspond with Stanton may reach him at 5908 Orleans Place, Alexandria, LA 71303. Or phone him at (318)442-0449, home; (318)443-3279, work. Readers may send Stanton a blank diskette and SASE return mailer with $1 and he will provide the program with improved documentation.

Typographical error spotted in pinout

Richard Ohi, who provided a User Note in April about the Smith Corona Messenger Module, noticed a typographical error in the pinouts that were published. In the pinout listing, pin No. 1 on the TI parallel port was inadvertently shown connected to pins 1 and 2 on the Centronics connector. The correct connection is pin 1 to pin 1 and pin 2 to pin 2.

Control characters are needed in Palette Master

We lied. Last month's we said the blank spaces in lines 110 and 120 of the Palette Master program from February are supposed to be blank. Not so, according to several astute readers, including Jim Uzzell of Austin, Texas. Actually, the spaces are occupied by control characters that cannot be seen when listing the program to the screen. Once again, here are the two lines:

110 DISPLAY AT(5,20):" "

Between the quotes, enter CTRL H and CTRL J.

120 DISPLAY AT(6,20):" "

Between the quotes, enter CTRL I and CTRL K.

To check these lines, after entering the control characters, place a REM in front of each line and list them. In place of the control characters in line 110 you should see reserved works RETURN DIM. In place of the control characters in line 120 you should see DEF END.

Re-ink printer ribbons

This comes from James W. Wiegand of Ridgeley, West Virginia. he writes:

I have a solution to the problem of finding free ribbons for printers such as the Seikosha GP-100TI, Gorilla Banana, Radio Shack DMP 100 and others. These all use a two-cartridge, continuous loop ribbon that is hard to find and generally not very fresh. After repeated replacements, I decided that there had to be a way to re-ink these cartridges. The sketches below show a modification which has allowed me to use the same ribbon for over a year, printing many pages of documentation.

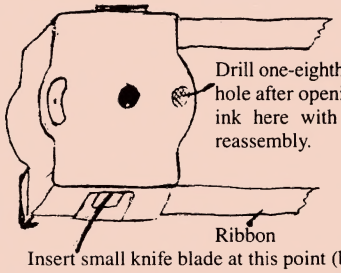
The modification is simple and requires only a one-eighth inch drill bit to accomplish.

I suggest experimenting with an old ribbon first. Start by carefully separating the case on the left side, as installed in the printer. Note as it comes apart how the ribbon and one small gear are positioned. In the area shown in Sketch A, drill a small hole through the top half of the case. This hole will be used to inject ink onto the felt wheel inside. A medicine dropper works fine for me.

Noting Sketch B, reassemble the cartridge. Make sure that the ribbon is not snagged by the case.

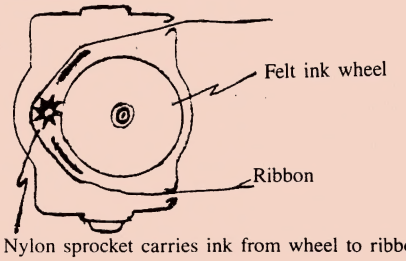
Install your modified ribbon assembly in the printer and put the printer in a self-test mode. As the head moves across the paper you should be able to see the felt wheel moving. With your injector, apply black stamp-pad ink to the wheel. Do not put ink directly onto the ribbon, or you might have a major cleaning operation. There are inks available for use with printer ribbons, but I have had not problem with the stamp pad ink.

Sketch A



Insert small knife blade at this point (both sides) and gently pry case open.

Sketch B



Nylon sprocket carries ink from wheel to ribbon.

MICROpendium pays $10 for items used in *User Notes*. Mail to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680.

1989 FAIRS

MARCH

West Coast Computer Fair, March 17-19, Brooks Hall, San Francisco. San Francisco 99ers to be at Booth 733. For further information, write San Francisco 99ers, 24816 Mango St., Hayward CA 94545.

TICOFF (TI Computer Owners Fun Faire), March 18, Roselle Park High School, Roselle Park, New Jersey. Write TICOFF'89 c/o Roselle Park High School, 185 West Webster Ave., Roselle Park, NJ 07204, or call Robert Guellnitz at (201) 241-4550 or (201) 382-5963 or the TICOFF BBS, (201) 241-8902.

APRIL

Fourth Annual New England TI Fayuh, April 1, Ramada Inn in Woburn, Massachusetts. Contact the Boston Computer Society TI99/4A User Group, One Center Plaza, Boston MA 02108.

Alberta TI-Orphan Reunion, April 29 at Innisfail Country Lodge, Innisfail, Alberta, Canada. Contact Fred Kessler, Box 20, Sundre, Alberta, Canada T0M 1X0 or (403) 638-3916.

4th Annual Ottawa TI-FEST, April 29 at Merivale High School in Nepean, Ontario, Canada. Contact Jane Laflamme, 5480 Canotek Rd. Unit #10, Gloucester, Ontario, Canada K1J 9H6 or (613) 745-2225.

MAY

Multi User Group Conference May 20, Reed Hall/Student Activities Building, Ohio State University, Lima, Ohio. Write Lima Users Group, P.O. Box 647, Venedocia, OH 45894, or call Dave Szipple evenings at (419) 228-7109.

JUNE

TI99/4A Users Group (U.K.) Annual Meeting June 17 in Romley, England. For information, contact Stephen Shaw, 10 Alstone Rd., Stockport, Cheshire, England SK4 5AH.

SEPTEMBER

TI International Expo 89 Sept. 16 at Howard Johnson Inn, 5821 Richmond Highway, Alexandria, Virginia. For further information write Mid-Atlantic Ninety-Nners, TI International Expo 89, P.O. Box 4005, Rockville, MD 20850, (301) 340-7179; or Delphi TI-NET, Teledata; or CompuServe, 74405,1207.

OCTOBER

Fourth European Tref, begins at 10 a.m. Oct. 7 at Kolpinghuis, Nijmegen, The Netherlands. For information, contact Berry Harmsen, 1e, Oosterparstr 141e, 1091 GZ, Amsterdam, Holland.

Australia TI Fair, 2-6 p.m. Oct. 14, Pavilion, Deepdene Park, Whitehorse Rd., Deepdene, Australia. For information contact TI99/4A Users Group — Melbourne Inc., 88 Main St., Blackburn, Victoria 3130, Australia.

3rd International TI-Users Meeting, 10 a.m.-6 p.m. Oct. 15 at Jugenderherberge Duisberg Wedau, Kalkweg 148, 4100 Duisberg 48, West Germany. For information contact TI-99er Workshop Rheinland, Dept. Allgemein & Software, c/o Mike Heuser, Karl-Marx-Allee 18, 5000 Cologne 71, West Germany, or the organizing committee at PCC, TI-Service, c/o Hans Greiffenberg, Großglocknerstr. 45, D-4100 Duisberg 28., West Germany.

Third Annual CPUG Computer Expo, 7 a.m.-2 p.m. Oct. 15 at Carlisle Fairgrounds on Clay Street in Carlisle, Pennsylvania. Sponsored by Central Pennsylvania 99/4A Users Group, co-sponsored by Cumberland County Amateur Radio Service and 6th Annual Cumberland County Hamfest. For information, contact Central Pennsylvania 99/4A Users Group, P.O. Box 14126, Harrisburg, PA 17104-0126 or the WIZ/TIB BBS, (717) 657-4992 or 657-4997.

NOVEMBER

Chicago TI-Faire, 9 a.m.-5 p.m. Nov. 4 at Holiday Inn, 3505 Algonquin Rd., Rolling Meadows, Illinois. Social evening Nov. 3, dinner evening of Nov. 4. Sponsored by Chicago Area TI99/4A Users Group. Contact Sandy Bartels, Chicago Area TI99/4A Users Group, P.O. Box 578341, Chicago, IL 60657 or (312) 859-3850.

Milwaukee TI-Faire, 9 a.m.-5 p.m. Nov. 5 at Quality Inn, 5311 S. Howell Ave., Milwaukee, Wisconsin (across from Mitchell Field Airport). For information call Gene Hitz, 4122 N. Glenway, Milwaukee, WI 53222 or (414) 535-0133.

TI FAIR '90

FEBRUARY

TI-Fest West '90, Feb. 17-18, Day's Inn, 88 E. Broadway, Tucson, Arizona. Sponsored by Southwest 99ers. For information, call (602) 747-5046 or the Cactus Patch BBS, (602) 795-1953 or check GEnie. For room reservations, call (602) 791-7581 by Jan. 16 and mention Fest-West.

This TI event listing is a permanent feature of MICROpendium. User groups and others planning events for TI/Geneve users may send information for inclusion in this standing column. Events will remain listed throughout the year for reference for the coming year.

Classified

HARDWARE

FOR SALE

TI external drive with power supply (PHP1850) $60, TI internal drive (PHP1250), TI32K card (PHP1260) $85, TI cassette recorder (PHP2700) $25, TI video modulator (UM1381) $5, TI joysticks (PHP1100) $2, TI parallel printer cables (PHP2500) $10, dual cassette recorder cables $4, Cassette Software ($2 ea): Teach Yourself Extended BASIC, Personal Financial Aids; Disk Software ($5 ea): Text to Speech-English; Cartridge Software ($3 ea): Blasto, Jawbreakerll, Tombstone City, Super Demon Attack, Hunt the Wumpus, TI Invaders, Hopper, Parsec, Chisholm Trail, Car Wars, Munchman, The Attack, Football, Picnic Paranoia, Alpiner, Munchmobile, Return to Pirates Isle; ($5 ea): Personal Record Keeping, Personal Report Generator, Home Financial Decisions, Household Budget Management, Tax/Investment Record Keeping, Adventure cartridge with Pirate Adventure on cassette. Call Terry at 312-425-0317 eves after 6:30 PM CDST best, all day weekends. v6n5

SOFTWARE

ROLL CREDITS

Announcing ROLL CREDITS with a special price and a free program with your order. Video "movie" makers, give that special touch to your productions. Create professional titles with the Roll Credits title generator. Only $7 for cassette or $9.95 for diskette. Send SASE for details or check with your order to: Newcastle Enterprises, 13424 North 33rd Place, Phoenix, AZ 85046.

ASSEMBLY LANGUAGE MUSIC

Plays under Extended BASIC. Johann Sebastian Bach's Goldberg Variations (Aria with thirty variations). Plays about fifty minutes. Requires XB, 32K, and disk drive. Will not execute correctly on Geneve or "bus" modified systems. Send $4.50 to Harrison Software, 5705 40th Place, Hyattsville, MD, 20781. v6n5

"BOOT PROGRAM"

Copyrighted by the Miami Users Group Feb. 1989. Not available from any other source or Mail Order Co.

Latest up-to-date version by the original author, John Johnson.

"BOOT" is in assembly language and uses the Horizon RAM Disk "MENU" program, without a RAM Disk.

You will be able to:

1. View a file
2. Catalog a disk
3. Run a program, XB, EA, or cartridges.
4. Access any program on any drive
5. Even run XB programs over 42 sectors.
6. Print any file or disk catalog.

Send $7.95 to:

MIAMI USERS GROUP
YVETTE MCKENZIE
6775 TAMIAMI CANAL ROAD
MIAMI, FL 33126

v6n5

Classified

SOFTWARE

FOR SALE

120 original programs $1 each, 18 full collection disks $5 each, 5 Tips disks reduced to $5 each, 3 Nuts & Bolts disks reduced to $10 each. Send $1 refundable for Tigercub catalog. 250 disks of public domain and fairware, $1.50 each. Send SASE for list or $1 refundable) for catalog, to TI-PD, 156 Collingwood Ave., Whitehall OH 43213. v6n12

CBM-III

Now in V5.6 — a better mousetrap for your finances. More than 50 menu-driven functions (recording, editing, balancing, reconciling, sorting, searching, analysis, etc.) with 32 help screens from a 4-screen menu. On-disk documentation. Epson printer compatible. Geneve compatible. Specify greatest disk drive configuration. $15.00 to: W. Irving Crowley, Lost Canyon Rd., Pine Level, AL 36065. Ph (205) 584-7644. v6n6

SYSTEMS

TI99/4A

Speech Synth, PEB, monitor, Panasonic KX-P1091, XB, TI-Writer, Disk Fixer, Disk Manager, SS, drive, 128K, RS232, 17 cartridges, with desk and printer stand, make offer. K. Marinelli, Beverly, MA (508) 927-6928. v6n6

FOR SALE

Geneve 9640, enhanced keyboard, mouse, disks, manuals, $375. John Norcross, B Co., 26th Signal Bn. APO NY 09176. v6n5

WANTED

WANTED

Volume 5, Number 1 of Home Computer Magazine, and Volume 2, Number 3 of Enthusiast 99 (May/June 1984 issue). Must be in excellent condition. Will pay $6 for HCM and $10 for Enthusiast 99. I will also reimburse postage costs. Bill Gaskill, Box 2642, Grand Junction, Co. 81502 303-242-8842. v6n5

WANTED TO BUY

TI99/4A Mario Brothers Game. David Hall, 2703 Beaver Creek Crossing, Powder Springs, Ga. 30073. (404) 943-1568. v6n5

BOOK WANTED

Introduction to the USCD p-System by Charles W. Grant and Jon Butah, published by SYBEX. Write or phone Ed Livingston, 244 Walt Arney Rd., Lenoir, North Carolina 28645 (704) 754-4089. v6n5

MISCELLANEOUS

Texas Instruments
TI-99/4a
Hardware & Software

—TURBO-PASC'99 (Pascal)! $59.95 E/A required.—
Proto-Board for PEB by Willforth only $35. Joy-Sticks $10.00 XB Detective $10.00. Rave '99 KeyBoard!! $199. Speech Editor $20. Printer Cable (PIO) $24. Jungle Hunt, Protector II, Moon Patrol, PacMan $15 ea. TI Dust Cover $4, Adventure $10.00. Adv Hint Book $8. TI Extended BASIC $40, P-Code Card w/software $159. TI-Artist $20. Princess & Frog, AntEater, HenPecked $10 ea.

*TI-BASE $25 (great) Data Base, PEB Hardware Manual 32K, Disk Ctrl, RS232, RAMdisk Cards, Cables & Connectors

L.L. CONNER ENTERPRISE
COMPUTER&ELECTRONICS
1521 Ferry Street
Lafayette, IN 47904

CALL 317-742-8146 or CALL 317-423-4879 to order UPS today!
Visa or M/C Accepted, add $3 shipping v6n6

Policy

The cost of classified advertising is 25 cents per word. Classified display (i.e., special formatting or graphics) is $9 per column inch. Classified advertisements must be paid in advance. Classified advertisers may request a category under which they would like their advertisements to appear, but the final placement decision is the responsibility of the publisher.

Classified deadlines will be kept open for as long as practical. For the purpose of classified advertising deadlines, any classified ad received later than the first day of any month cannot be assured of placement in the next edition. We will do our best to include every advertisement that is submitted in the earliest possible edition.

The publisher offers no guarantee that any advertisement will be published in any particular issue. Any damages that result either from errors in copy or for failure to be included in any particular edition will be limited to the amount of the cost of the advertisement itself. The publisher reserves the right to reject any advertisement.

The advertiser may elect to publish the advertisement in subsequent editions at the same charge, payable prior to publication. The deadline for carryover classifoeds is the same as for new advertising.

In submitting an ad, please indicate whether you would like a refund if it is not published in the requested edition or whether you would like us to hold it for the next edition. Cancellations and refunds cannot be made after the second day of the month.

Send classified advertising to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680.

MISCELLANEOUS

PASCAL COMPLETE SYSTEM	$150
PASCAL CARD ONLY	$ 80
FULL TI PE/BOX	$350
EMPTY TI PE/BOX	$135
18" P-BOX EXTENSION CABLE	$ 25
TI99/4A new sealed in box	$110
SPEECH SYNTHESIZER used	$ 45
PARALLEL PRINTER CABLE 6'	$ 20
WORD WRITER + W/PARALLEL CABLE	$ 49
STAND ALONE DISK DRIVE "TI"	$ 79
MYARC HARD CONTROLLER CARD new	$299
TI ORIGINAL COLOR MONITOR	$175
SERVICE MANUAL (CONSOLE/P BOX)	$ 25
TI99/4A COMPUTER used w/books	$ 55
EXTENDED BASIC used w/new book	$ 35

PLUS 5% SHIPPING CALL OR WRITE for complete free list
JIM LESHER, 722 HUNT DALLAS, TEXAS 75214, 214 821 9274 v6,n5

ENORMOUS INVENTORY

TI 99/4A. Laser,
Apple & IBM Compatibles.

"Some Commodore and Atari Items. (No Catalog on Above Two Systems.)" Full Hardware & Software on All Other Systems. Complete Line of Computer Forms.

BRAATZS
Computer Services

719 E. Byrd St., Appleton, WI 54911. **Catalogs** $2. MC/Visa accepted. 414-731-3478 (ORDER LINE ONLY) (Call 414-731-4320 after 6:00 P.M. Wisconsin time.) Custom made covers (send sizes). v6n6

The LEADING monthly devoted to the TI99/4A

Subscription Fees

$20 for 12 issues via domestic second class mail
$25.25 (U.S. funds) Mexican delivery
$27.50 (U.S. funds) Canadian delivery
$25.00 (U.S. funds) for 12 issues other foreign delivery via surface mail
$37.00 (U.S. funds) for 12 issues other foreign delivery via air mail

Outside U.S., pay via postal or international money order or credit card; personal checks from non-U.S. banks will be returned

Address Changes

Subscribers who move may have the delivery of their most recent issue(s) delayed unless MICROpendium is notified six weeks in advance of address changes. Please include your old address as it appears on your mailing label when making an address change.

Back Issue Policy

Back issues of MICROpendium are available to subcribers only. Those wishing back issues may notify us of the issue(s) desired and include $2 per issue desired in a check or money order or by credit card. (Minimum credit card order is $9.) No shipping charge in U.S. and Mexico; Texas residents add 7.5% sales tax. Shipping charge of 30 cents per issue to Canada. For other foreign delivery, add 50 cents per issue surface mail, $2 per issue air mail. No discounts on orders of sets. All prices U.S. funds.

OUT OF STOCK: Vol 1, nos. 1-2, Vol 2, no. 1

Miscellany

TI-Forth Manual (Program and demo disks included).....$20.00
MICROpendium Index (2-SSSD disks, XBASIC required).....$5.00
Disk of programs from one issue of MICROpendium (must be a subscriber to order).....$4.00
12 monthly disks of programs appearing in each edition of MICROpendium (must be a subscriber to order/starts with April 1989).....$40.00
Magazine holders (12/set-add $1 shipping/order).....$3.00

Send name, address, product(s) ordered, check, money order or Visa/MasterCard number and expiration date ($9 minimum on credit card orders $9) to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680. (Foreign orders write for postage fees.)

Tell us about it

Please let us what columns or features you like the most about MICROpendium. Rank your selections in order of preference using this form. Return it to us when you renew your subscription.

1. _____
2. _____
3. _____
4. _____

Other suggestions:_____

_____

_____

Send me the next 12 issues of MICROpendium. I am enclosing $ in a check or money order in U.S. funds. Or bill my ☐ ☐ (Check one)

Exp. Date _____

Card No. _____

Minimum credit card order is $9

Signature_____

(required on credit card orders)

Mail to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680

Name_____

Address_____

City_____

State____**ZIP**_____

The numbers on the left of your mailing label indicates the cover date of the last issue on your subscription. v6.n5

SECOND CLASS